01-05-00

Practitioner's Docket No. L09-99-048 **PATENT**

A

jc610 U.S. PTO
01/04/00

Preliminary Classification:

Proposed Class:

Subclass:

NOTE: "All applicants are requested to include a preliminary classification on newly filed patent applications. The preliminary classification, preferably class and subclass designations, should be identified in the upper right-hand corner of the letter of transmittal accompanying the application papers, for example 'Proposed Class 2, subclass 129.'" M.P.E.P. § 601, 7th ed.

jc675 U.S. PTO
09/477477
01/04/00

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

Box Patent Application
Assistant Commissioner for Patents
Washington, D.C. 20231

NEW APPLICATION TRANSMITTAL

Transmitted herewith for filing is the patent application of

Inventor(s): Estrada, Julio, Estrada, Miguel A., Russo, Joseph A., Shore, Maurice B.

WARNING: *37 C.F.R. § 1.41(a)(1) points out:*

"(a) A patent is applied for in the name or names of the actual inventor or inventors.

"(1) The inventorship of a nonprovisional application is that inventorship set forth in the oath or declaration as prescribed by § 1.63, except as provided for in § 1.53(d)(4) and § 1.63(d). If an oath or declaration as prescribed by § 1.63 is not filed during the pendency of a nonprovisional application, the inventorship is that inventorship set forth in the application papers filed pursuant to § 1.53(b), unless a petition under this paragraph accompanied by the fee set forth in § 1.17(i) is filed supplying or changing the name or names of the inventor or inventors."

For (title):
SYSTEM AND METHOD FOR DYNAMICALLY GENERATING VIEWABLE GRAPHICS

CERTIFICATION UNDER 37 C.F.R. § 1.10*
(Express Mail label number is mandatory.)
(Express Mail certification is optional.)

I hereby certify that this New Application Transmittal and the documents referred to as attached therein are being deposited with the United States Postal Service on this date January 4, 2000, in an envelope as "Express Mail Post Office to Addressee," mailing Label Number EH342805183US, addressed to the: Assistant Commissioner for Patents, Washington, D.C. 20231.

Elizabeth A. Hart
(type or print name of person mailing paper)

Elizabeth A. Hart
Signature of person mailing paper

WARNING: *Certificate of mailing (first class) or facsimile transmission procedures of 37 C.F.R. § 1.8 cannot be used to obtain a date of mailing or transmission for this correspondence.*

"WARNING: *Each paper or fee filed by "Express Mail" **must** have the number of the "Express Mail" mailing label placed thereon prior to mailing. 37 C.F.R. § 1.10(b).*
*"Since the filing of correspondence under § 1.10 without the Express Mail mailing label thereon is an oversight that can be avoided by the exercise of reasonable care, requests for waiver of this requirement will **not** be granted on petition." Notice of Oct. 24, 1996, 60 Fed. Reg. 56,439, at 56,442.*

1. Type of Application

This new application is for a(n)

(check one applicable item below)

- [X] Original (nonprovisional)
- [ ] Design
- [ ] Plant

WARNING: *Do not use this transmittal for a completion in the U.S. of an International Application under 35 U.S.C. § 371(c)(4), unless the International Application is being filed as a divisional, continuation or continuation-in-part application.*

WARNING: *Do not use this transmittal for the filing of a provisional application.*

NOTE: If one of the following 3 items apply, then complete and attach ADDED PAGES FOR NEW APPLICATION TRANSMITTAL WHERE BENEFIT OF A PRIOR U.S. APPLICATION CLAIMED and a NOTIFICATION IN PARENT APPLICATION OF THE FILING OF THIS CONTINUATION APPLICATION.

- [ ] Divisional.
- [ ] Continuation.
- [ ] Continuation-in-part (C-I-P).

2. Benefit of Prior U.S. Application(s) (35 U.S.C. §§ 119(e), 120, or 121)

NOTE: A nonprovisional application may claim an invention disclosed in one or more prior filed copending nonprovisional applications or copending international applications designating the United States of America. In order for a nonprovisional application to claim the benefit of a prior filed copending nonprovisional application or copending international application designating the United States of America, each prior application must name as an inventor at least one inventor named in the later filed nonprovisional application and disclose the named inventor's invention claimed in at least one claim of the later filed nonprovisional application in the manner provided by the first paragraph of 35 U.S.C. § 112. Each prior application must also be:

(i) An international application entitled to a filing date in accordance with PCT Article 11 and designating the United States of America; or

(ii) Complete as set forth in § 1.51(b); or

(iii) Entitled to a filing date as set forth in § 1.53(b) or § 1.53(d) and include the basic filing fee set forth in § 1.16; or

(iv) Entitled to a filing date as set forth in § 1.53(b) and have paid therein the processing and retention fee set forth in § 1.21(l) within the time period set forth in § 1.53(f).

37 C.F.R. § 1.78(a)(1).

NOTE: If the new application being transmitted is a divisional, continuation or a continuation-in-part of a parent case, or where the parent case is an International Application which designated the U.S., or benefit of a prior provisional application is claimed, then check the following item and complete and attach ADDED PAGES FOR NEW APPLICATION TRANSMITTAL WHERE BENEFIT OF PRIOR U.S. APPLICATION(S) CLAIMED.

WARNING: *If an application claims the benefit of the filing date of an earlier filed application under 35 U.S.C. §§ 120, 121 or 365(c), the 20-year term of that application will be based upon the filing date of the earliest U.S. application that the application makes reference to under 35 U.S.C. §§ 120, 121 or 365(c). (35 U.S.C. § 154(a)(2) does not take into account, for the determination of the patent term, any application on which priority is claimed under 35 U.S.C. §§ 119, 365(a) or 365(b).) For a c-i-p application, applicant should review whether any claim in the patent that will issue is supported by an earlier application and, if not, the applicant should consider canceling the reference to the earlier filed application. The term of a patent is not based on a claim-by-claim approach. See Notice of April 14, 1995, 60 Fed. Reg. 20,195, at 20,205.*

WARNING: *When the last day of pendency of a provisional application falls on a Saturday, Sunday, or Federal holiday within the District of Columbia, any nonprovisional application claiming benefit of the provisional application* ***must*** *be filed prior to the Saturday, Sunday, or Federal holiday within the District of Columbia. See 37 C.F.R. § 1.78(a)(3).*

☐ The new application being transmitted claims the benefit of prior U.S. application(s). Enclosed are ADDED PAGES FOR NEW APPLICATION TRANSMITTAL WHERE BENEFIT OF PRIOR U.S. APPLICATION(S) CLAIMED.

3. Papers Enclosed

A. Required for filing date under 37 C.F.R. § 1.53(b) (Regular) or 37 C.F.R. § 1.153 (Design) Application

102 Pages of specification

9 Pages of claims

33 Sheets of drawing

WARNING: ***DO NOT*** *submit original drawings. A high quality copy of the drawings should be supplied when filing a patent application. The drawings that are submitted to the Office must be on strong, white, smooth, and non-shiny paper and meet the standards according to § 1.84. If corrections to the drawings are necessary, they should be made to the original drawing and a high-quality copy of the corrected original drawing then submitted to the Office. Only one copy is required or desired. For comments on proposed then-new 37 C.F.R. § 1.84, see Notice of March 9, 1988 (1990 O.G. 57-62).*

NOTE: "Identifying indicia, if provided, should include the application number or the title of the invention, inventor's name, docket number (if any), and the name and telephone number of a person to call if the Office is unable to match the drawings to the proper application. This information should be placed on the back of each sheet of drawing a minimum distance of 1.5 cm. (5/8 inch) down from the top of the page . . ." 37 C.F.R. § 1.84(c)).

(complete the following, if applicable)

☐ The enclosed drawing(s) are photograph(s), and there is also attached a "PETITION TO ACCEPT PHOTOGRAPH(S) AS DRAWING(S)." 37 C.F.R. § 1.84(b).

☐ formal

☒ informal

B. Other Papers Enclosed

8 Pages of declaration and power of attorney

1 Pages of abstract

____ Other

4. Additional papers enclosed

☐ Amendment to claims

☐ Cancel in this applications claims _____ before calculating the filing fee. (At least one original independent claim must be retained for filing purposes.)

☐ Add the claims shown on the attached amendment. (Claims added have been numbered consecutively following the highest numbered original claims.)

☐ Preliminary Amendment

☐ Information Disclosure Statement (37 C.F.R. § 1.98)

☐ Form PTO–1449 (PTO/SB/08A and 08B)

☐ Citations

☐ Declaration of Biological Deposit

☐ Submission of "Sequence Listing," computer readable copy and/or amendment pertaining thereto for biotechnology invention containing nucleotide and/or amino acid sequence.

☐ Authorization of Attorney(s) to Accept and Follow Instructions from Representative

☐ Special Comments

☐ Other

5. Declaration or oath (including power of attorney)

NOTE: A newly executed declaration is not required in a continuation or divisional application provided that the prior nonprovisional application contained a declaration as required, the application being filed is by all or fewer than all the inventors named in the prior application, there is no new matter in the application being filed, and a copy of the executed declaration filed in the prior application (showing the signature or an indication thereon that it was signed) is submitted. The copy must be accompanied by a statement requesting deletion of the names of person(s) who are not inventors of the application being filed. If the declaration in the prior application was filed under § 1.47, then a copy of that declaration must be filed accompanied by a copy of the decision granting § 1.47 status or, if a nonsigning person under § 1.47 has subsequently joined in a prior application, then a copy of the subsequently executed declaration must be filed. See 37 C.F.R. §§ 1.63(d)(1)–(3).

NOTE: A declaration filed to complete an application must be executed, identify the specification to which it is directed, identify each inventor by full name including family name and at least one given name, without abbreviation together with any other given name or initial, and the residence, post office address and country or citizenship of each inventor, and state whether the inventor is a sole or joint inventor. 37 C.F.R. § 1.63(a)(1)–(4).

NOTE: "The inventorship of a nonprovisional application is that inventorship set forth in the oath or declaration as prescribed by § 1.62, except as provided for in § 1.53(d)(4) and § 1.63(d). If an oath or declaration as prescribed by § 1.63 is not filed during the pendency of a nonprovisional application, the inventorship is that inventorship set forth in the application papers filed pursuant to § 1.53(b), unless a petition under this paragraph accompanied by the fee set forth in § 1.17(i) is filed supplying or changing the name or names of the inventor or inventors." 37 C.F.R. § 1.41(a)(1).

☒ Enclosed

Executed by

(check **all** *applicable boxes)*

☐ inventor(s).

☐ legal representative of inventor(s). 37 C.F.R. §§ 1.42 or 1.43.

☐ joint inventor or person showing a proprietary interest on behalf of inventor who refused to sign or cannot be reached.

☐ This is the petition required by 37 C.F.R. § 1.47 and the statement required by 37 C.F.R. § 1.47 is also attached. *See* item 13 below for fee.

☐ Not Enclosed.

NOTE: Where the filing is a completion in the U.S. of an International Application or where the completion of the U.S. application contains subject matter in addition to the International Application, the application may be treated as a continuation or continuation-in-part, as the case may be, utilizing ADDED PAGE FOR NEW APPLICATION TRANSMITTAL WHERE BENEFIT OF PRIOR U.S. APPLICATION CLAIMED.

☐ Application is made by a person authorized under 37 C.F.R. § 1.41(c) on behalf of *all* the above named inventor(s).

(The declaration or oath, along with the surcharge required by 37 C.F.R. § 1.16(e) can be filed subsequently).

☐ Showing that the filing is authorized.
(not required unless called into question. 37 C.F.R. § 1.41(d))

6. Inventorship Statement

WARNING: *If the named inventors are each not the inventors of all the claims an explanation, including the ownership of the various claims at the time the last claimed invention was made, should be submitted.*

The inventorship for all the claims in this application are:

☒ The same.

or

☐ Not the same. An explanation, including the ownership of the various claims at the time the last claimed invention was made,

☐ is submitted.

☐ will be submitted.

7. Language

NOTE: An application including a signed oath or declaration may be filed in a language other than English. An English translation of the non-English language application and the processing fee of $130.00 required by 37 C.F.R. § 1.17(k) is required to be filed with the application, or within such time as may be set by the Office. 37 C.F.R. § 1.52(d).

☒ English

☐ Non-English

☐ The attached translation includes a statement that the translation is accurate. 37 C.F.R. § 1.52(d).

8. Assignment

☐ An assignment of the invention to _____

_____

☐ is attached. A separate ☐ "COVER SHEET FOR ASSIGNMENT (DOCUMENT) ACCOMPANYING NEW PATENT APPLICATION" or ☐ FORM PTO 1595 is also attached.

☒ will follow.

NOTE: "If an assignment is submitted with a new application, send two separate letters-one for the application and one for the assignment." Notice of May 4, 1990 (1114 O.G. 77-78).

WARNING: *A newly executed "CERTIFICATE UNDER 37 C.F.R. § 3.73(b)" must be filed when a continuation-in-part application is filed by an assignee. Notice of April 30, 1993, 1150 O.G. 62-64.*

9. Certified Copy

Certified copy(ies) of application(s)

Country	Appln. No.	Filed
Country	Appln. No.	Filed
Country	Appln. No.	Filed

from which priority is claimed

☐ is (are) attached.

☐ will follow.

NOTE: The foreign application forming the basis for the claim for priority must be referred to in the oath or declaration. 37 C.F.R. § 1.55(a) and 1.63.

NOTE: This item is for any foreign priority for which the application being filed directly relates. If any parent U.S. application or International Application from which this application claims benefit under 35 U.S.C. § 120 is itself entitled to priority from a prior foreign application, then complete item 18 on the ADDED PAGES FOR NEW APPLICATION TRANSMITTAL WHERE BENEFIT OF PRIOR U.S. APPLICATION(S) CLAIMED.

10. Fee Calculation (37 C.F.R. § 1.16)

A. ☒ Regular application

CLAIMS AS FILED						
Number filed		Number Extra		Rate	Basic Fee 37 C.F.R. § 1.16(a) $760.00	
Total Claims (37 C.F.R. § 1.16(c))	13	− 20 =	0	×	$ 18.00	0
Independent Claims (37 C.F.R. § 1.16(b))	7	− 3 =	4	×	$ 78.00	312
Multiple dependent claim(s), if any (37 C.F.R. § 1.16(d))				+	$260.00	

☐ Amendment cancelling extra claims is enclosed.

☐ Amendment deleting multiple-dependencies is enclosed.

☐ Fee for extra claims is not being paid at this time.

NOTE: If the fees for extra claims are not paid on filing they must be paid or the claims cancelled by amendment, prior to the expiration of the time period set for response by the Patent and Trademark Office in any notice of fee deficiency. 37 C.F.R. § 1.16(d).

Filing Fee Calculation $ 1,072

B. ☐ Design application ($310.00—37 C.F.R. § 1.16(f))

Filing Fee Calculation $_____

C. ☐ Plant application
($480.00—37 C.F.R. § 1.16(g))

Filing fee calculation $_____

11. Small Entity Statement(s)

☐ Statement(s) that this is a filing by a small entity under 37 C.F.R. § 1.9 and 1.27 is (are) attached.

WARNING: *"Status as a small entity must be specifically established in each application or patent in which the status is available and desired. Status as a small entity in one application or patent does not affect any other application or patent, including applications or patents which are directly or indirectly dependent upon the application or patent in which the status has been established. The refiling of an application under § 1.53 as a continuation, division, or continuation-in-part (including a continued prosecution application under § 1.53(d)), or the filing of a reissue application requires a new determination as to continued entitlement to small entity status for the continuing or reissue application. A nonprovisional application claiming benefit under 35 U.S.C. § 119(e), 120, 121, or 365(c) of a prior application, or a reissue application may rely on a statement filed in the prior application or in the patent if the nonprovisional application or the reissue application includes a reference to the statement in the prior application or in the patent or includes a copy of the statement in the prior application or in the patent and status as a small entity is still proper and desired. The payment of the small entity basic statutory filing fee will be treated as such a reference for purposes of this section." 37 C.F.R. § 1.28(a)(2).*

WARNING: *"Small entity status must not be established when the person or persons signing the . . . statement can **unequivocally** make the required self-certification." M.P.E.P., § 509.03, 6th ed., rev. 2, July 1996 (emphasis added).*

(complete the following, if applicable)

☐ Status as a small entity was claimed in prior application

_____ / _____, filed on _____, from which benefit is being claimed for this application under:

35 U.S.C. § ☐ 119(e),
☐ 120,
☐ 121,
☐ 365(c),

and which status as a small entity is still proper and desired.

☐ A copy of the statement in the prior application is included.

Filing Fee Calculation (50% of **A, B** or **C** above)

$_____

NOTE: Any excess of the full fee paid will be refunded if small entitiy status is established and a refund request are filed within 2 months of the date of timely payment of a full fee. The two-month period is not extendable under § 1.136. 37 C.F.R. § 1.28(a).

12. Request for International-Type Search (37 C.F.R. § 1.104(d))

(complete, if applicable)

☐ Please prepare an international-type search report for this application at the time when national examination on the merits takes place.

13. Fee Payment Being Made at This Time

☐ Not Enclosed

☐ No filing fee is to be paid at this time.
(This and the surcharge required by 37 C.F.R. § 1.16(e) can be paid subsequently.)

☒ Enclosed

☒ Filing fee	$ 1,072
☐ Recording assignment ($40.00; 37 C.F.R. § 1.21(h)) (See attached "COVER SHEET FOR ASSIGNMENT ACCOMPANYING NEW APPLICATION".)	$ _____
☐ Petition fee for filing by other than all the inventors or person on behalf of the inventor where inventor refused to sign or cannot be reached ($130.00; 37 C.F.R. §§ 1.47 and 1.17(i))	$ _____
☐ For processing an application with a specification in a non-English language ($130.00; 37 C.F.R. §§ 1.52(d) and 1.17(k))	$ _____
☐ Processing and retention fee ($130.00; 37 C.F.R. §§ 1.53(d) and 1.21(l))	$ _____
☐ Fee for international-type search report ($40.00; 37 C.F.R. § 1.21(e))	$ _____

NOTE: 37 C.F.R. § 1.21(l) establishes a fee for processing and retaining any application that is abandoned for failing to complete the application pursuant to 37 C.F.R. § 1.53(f) and this, as well as the changes to 37 C.F.R. §§ 1.53 and 1.78(a)(1), indicate that in order to obtain the benefit of a prior U.S. application, either the basic filing fee must be paid, or the processing and retention fee of § 1.21(l) must be paid, within 1 year from notification under § 53(f).

Total fees enclosed $ 1,072

14. Method of Payment of Fees

☐ Check in the amount of $_____

☒ Charge Account No. DA-12 2158 in the amount of $ 1,072.

A duplicate of this transmittal is attached.

NOTE: Fees should be itemized in such a manner that it is clear for which purpose the fees are paid. 37 C.F.R. § 1.22(b).

15. Authorization to Charge Additional Fees

WARNING: *If no fees are to be paid on filing, the following items should not be completed.*

WARNING: *Accurately count claims, especially multiple dependent claims, to avoid unexpected high charges, if extra claim charges are authorized.*

☒ The Commissioner is hereby authorized to charge the following additional fees by this paper and during the entire pendency of this application to Account No. 08-1221 5 8 :

- ☒ 37 C.F.R. § 1.16(a), (f) or (g) (filing fees)
- ☒ 37 C.F.R. § 1.16(b), (c) and (d) (presentation of extra claims)

NOTE: *Because additional fees for excess or multiple dependent claims not paid on filing or on later presentation must only be paid or these claims cancelled by amendment prior to the expiration of the time period set for response by the PTO in any notice of fee deficiency (37 C.F.R. § 1.16(d)), it might be best not to authorize the PTO to charge additional claim fees, except possibly when dealing with amendments after final action.*

- ☒ 37 C.F.R. § 1.16(e) (surcharge for filing the basic filing fee and/or declaration on a date later than the filing date of the application)
- ☒ 37 C.F.R. § 1.17(a)(1)–(5) (extension fees pursuant to § 1.136(a)).
- ☒ 37 C.F.R. § 1.17 (application processing fees)

NOTE: *". . .A written request may be submitted in an application that is an authorization to treat any concurrent or future reply, requiring a petition for an extension of time under this paragraph for its timely submission, as incorporating a petition for extension of time for the appropriate length of time. An authorization to charge all required fees, fees under § 1.17, or all required extension of time fees will be treated as a constructive petition for an extension of time in any concurrent or future reply requiring a petition for an extension of time under this paragraph for its timely submission. Submission of the fee set forth in § 1.17(a) will also be treated as a constructive petition for an extension of time in any concurrent reply requiring a petition for an extension of time under this paragraph for its timely submission." 37 C.F.R. § 1.136(a)(3).*

- ☐ 37 C.F.R. § 1.18 (issue fee at or before mailing of Notice of Allowance, pursuant to 37 C.F.R. § 1.311(b))

NOTE: *Where an authorization to charge the issue fee to a deposit account has been filed before the mailing of a Notice of Allowance, the issue fee will be automatically charged to the deposit account at the time of mailing the notice of allowance. 37 C.F.R. § 1.311(b).*

NOTE: *37 C.F.R. § 1.28(b) requires "Notification of any change in status resulting in loss of entitlement to small entity status must be filed in the application . . . prior to paying, or at the time of paying, . . . the issue fee. . . " From the wording of 37 C.F.R. § 1.28(b), (a) notification of change of status must be made even if the fee is paid as "other than a small entity" and (b) no notification is required if the change is to another small entity.*

16. Instructions as to Overpayment

NOTE: ". . . Amounts of twenty-five dollars or less will not be returned unless specifically requested within a reasonable time, nor will the payer be notified of such amounts; amounts over twenty-five dollars may be returned by check or, if requested, by credit to a deposit account." 37 C.F.R. § 1.26(a).

☒ Credit Account No. DA-12 2158

☐ Refund

Express Mail Label No. EB 000 000 000 US

SIGNATURE OF PRACTITIONER

Reg. No. 34, 360

Stephen Keohane, Esq.

(type or print name of attorney)

Tel. No. (617) 693-4152

Lotus Development Corporation
55 Cambridge Parkway

P.O. Address

Customer No.

Cambridge, MA 02142

☐ **Incorporation by reference of added pages**

(check the following item if the application in this transmittal claims the benefit of prior U.S. application(s) (including an international application entering the U.S. stage as a continuation, divisional or C-I-P application) and complete and attach the ADDED PAGES FOR NEW APPLICATION TRANSMITTAL WHERE BENEFIT OF PRIOR U.S. APPLICATION(S) CLAIMED)

☐ Plus Added Pages for New Application Transmittal Where Benefit of Prior U.S. Application(s) Claimed

Number of pages added _____

☐ Plus Added Pages for Papers Referred to in Item 4 Above

Number of pages added _____

☐ Plus added pages deleting names of inventor(s) named in prior application(s) who is/are no longer inventor(s) of the subject matter claimed in this application.

Number of pages added _____

☐ Plus "Assignment Cover Letter Accompanying New Application"

Number of pages added _____

☐ **Statement Where No Further Pages Added**

(if no further pages form a part of this Transmittal, then end this Transmittal with this page and check the following item)

☒ This transmittal ends with this page.

SYSTEM AND METHOD FOR DYNAMICALLY GENERATING VIEWABLE GRAPHICS

BACKGROUND OF THE INVENTION

Cross References to Related Applications

Co-pending U.S. patent applications:

Serial Number 09/_____, entitled "System and Method for Interconnecting Secure Rooms", assignee docket LO999044;

Serial Number 09/_____, entitled "System and Method for Dynamic Management of Web Site", assignee docket LO999045;

Serial Number 09/_____, entitled "System and Method for Presentation of Room Navigation", assignee docket LO999046;

Serial Number 09/_____, entitled "System and Method for Independent Room Security Management", assignee docket LO99047;

Serial Number 09/_____, entitled "System and Method for Dynamic Browser Management of Web Site", assignee docket LO999049;

Serial Number 09/_____, entitled "System and Method for Room Decoration and Inheritance", assignee docket LO999051;

Serial Number 09/_____, entitled "System and Method for Online/Offline Uninterrupted Updating of Rooms in Collaboration Space", assignee docket LO999052;

Serial Number 09/_____, entitled "System and Method for Client Replication of Collaboration Space", assignee docket LO999053;

Serial Number 09/_____, entitled "System and Method for Browser Creation and Maintenance of Forms", assignee docket LO999054;

Serial Number 09/_____, entitled "System and Method for Browser Definition of Workflow Documents", assignee docket LO999055;

are assigned to the same assignee hereof and contain subject matter related, in certain respect, to the subject matter of the present application. The above-identified patent applications are incorporated herein by reference.

Technical Field of the Invention

This invention relates to web technology. More particularly, it relates to the creation and use of collaboration sites on the Internet or on an intranet client/server system and to the graphical user interface used in Internet communications.

Background Art

The Internet and the World Wide Web (WWW) provide intra-enterprise connectivity, inter-enterprise connectivity and application hosting on a larger scale than ever before. By exploiting the broadly available and deployed standards of the Internet and the WWW, system users and designers can leverage a single architecture to build client/server applications for internal use that can reach outside to customers, business partners and suppliers.

Collaboration requires simultaneous communication between individuals on a project team. Typically, this has required that the team members work in the same location. Phone and video conferencing has enabled some remote work on

the part of team members. Also, because of the growth of the Internet, collaboration using web technologies has been attempted, primarily using electronic mail (E-mail), Internet chat rooms, electronic whiteboards, and conferencing software. The most useful has been E-mail, but this approach results in a large trail or thread of notes as collaboration on a project advances, and these notes have no home or place to reside which is accessible by all team members substantially instantaneously and simultaneously. People often enter such a thread at different points, and such threads are not efficient in coordinating the work of many different people on a team which may include in-house developers and others, such as remote contractors, outside of an enterprise's firewall.

In order for such disperse teams to have the same, or substantially the same, collaboration environment as individuals working in the same physical office, a system is required which facilitates instant messaging, voice conferencing, electronic white boarding, and text and non-text file exchange. Such a system needs to provide a collaborative electronic room, or space, which is easily configured for use by team members without substantial administrative or application development support, and preferably include both groupware and project oriented

applications such as shared folders, file exchange, workflow, group calendars, threaded conversations, version control, file locking, file merging, and security.

There is a need in the art for such a system which is easy to set up and which enables diverse and remote teams to become immediately productive in a secure environment. It would be, further, most desirable to allow such a collaborative environment to be set up without administrative support, that is by members of the team itself, using a familiar and easy to use browser user interface. Members of the team, acting with manager or author authority, and using such a browser interface without involving administrative or application development support, need to be able to set up a folder or room for each project element, such as a source code component, with version control, workflow elements, and group calendaring for tracking the project or project element with respect to approvals and deadlines. Such a room needs to receive from team members reports and have them routed to appropriate team members for review, resolution, and approval.

Figure 1 shows a commonly used network arrangement in which a plurality of local computer systems 101-104 in a local area network (LAN) may access a plurality of remote

servers 105-108 through the Internet 100. Each remote server may be a web server (such as a Domino (TM) web server, available from Lotus Development Corporation of Cambridge, Massachusetts) for providing a web site for access by local computer systems 101-104. Each web site normally further provides a plurality of web pages to be served to the local computer systems upon request. Each local computer system may access the remote web sites with web browser software.

The WWW is a collection of servers on an IP (Internet Protocol) network, such as the Internet, an Intranet or an Extranet, that utilize the Hypertext Transfer Protocol (HTTP). Hereinafter, "Internet" 100 will be used to refer to any IP network.

HTTP is a known application protocol that provides users with access to files, which can be in different formats, such as text, graphics, images, sound, and video, using a standard page description language known as Hypertext Markup Language (HTML). Among a number of basic document formatting functions, HTML allows software developers to specify graphical pointers on displayed web pages, commonly referred to as "hyperlinks," that point to other web pages resident on remote servers. Hyperlinks

commonly are displayed as highlighted text or other graphical image on the web page. Selection of a hyperlink with a pointing device, such as a computer mouse, causes the local computer to download the HTML associated with the web page from a remote server. The browser then renders the HTML into the displayed web page.

Web pages accessed over the Internet, whether by a hyperlink, opening directly via an "open" button in the browser, or some other means, are commonly downloaded into the volatile cache of a local computer system. In a computer system, for example, the volatile cache is a high-speed buffer that temporarily stores web pages from accessed remote web sites. The volatile cache thus enables a user to quickly review web pages that were already downloaded, thereby eliminating the need to repeat the relatively slow process of traversing the Internet to access previously viewed web pages. This is called local caching.

On the server side, the first web servers were merely HTTP servers that resolved universal resource locators (URLs) by extracting literally from the URL the path to a file that contained the needed page, and transmitting the page back to the browser. Such a server was very simple; it could only be used to access static pages.

A "static" page is a page which, each time it is requested and served to a requestor, has the same byte content. That is, it does not depend upon which requestor is requesting the page, when the requestor is requesting the page, etc., the byte content of that page remains the same. By contrast, a "dynamic page" is a page which has byte content that may very well change depending upon the particular requestor, when the page is being requested, etc. This will be discussed further below.

It is important that web pages be served as quickly as possible, both to reduce the response time to a single user, and to increase the number of users that can be served concurrently. To improve the response time, the Web server uses caches. Web server caches are used to store web page responses in a readily accessible memory location so that when the web page is requested by a user, a previously cached web page response can be retrieved from cache and served quickly to the user.

Heretofore, collaboration on the Internet relied on the use of E-mail. The result has been the creation of a large thread or trail of notes having no home or place. It is a characteristic of such threads that people enter thread at

different points and may or may not have ready access to the information required to facilitate collaboration.

In past, there has been no convenient way to take information off a word processor document and put on the Internet in one step. To put a document file on the Internet for viewing, a user must establish a web server, load the file into a word processor, save it as HTML, and then find other related files and copy all of the files to the web server, and put them in the correct directory. One solution for this complex procedure is a web folder, that allows a user to upload files to Internet for viewing. Such a web folder presents a web server to the user as if it were a regular file directory, so the user can save the files to this directory. Even in this solution, it is still necessary for the user to establish the web server or obtain permission to use one already created. Further, this approach does not present context of document to the user during the upload process. Consequently, there is a need in the art for a way for a user to create a web site without web server, and create a document with pages within that site a created context, which is where on the Internet this document appears.

Previously, in a hierarchical database, security could

not be increased in subfiles with respect to a parent file. Also, in the past, directory entries or other indicia identifying objects to which a user did not have access were shown, but were greyed out or otherwise managed so that user access was inhibited. There is a need in the art, therefore, for providing for managing access to files in a hierarchical database where security may increase or decrease at a child with respect to its parent, and where indicia identifying objects to which the user does not have access not displayed.

Heretofore, a user can upload a document to a web site through the use of directories and file trees. These may not be intuitive for the user. Consequently, there is a need in the art for a way to upload documents which involves the more intuitive process of dragging and dropping the document to the spot on the site where it is to be rendered.

In accordance with various systems for supporting remote application execution, a Java version of a database engine is configured with a transactional replication feature by which there is brought down from a server to a browser, client data and application logic, or dynamic HTML is downloaded to a browser environment where developers can write a transportable application from scratch. However,

these systems, including the transactional replication feature, do not download the security model, and thus are not user filtered in the sense that access control lists, reader filtration, and selective replication are supported at the browser. Security, in such systems, is administered at the server, and not at the browser. Consequently, there is a need in the art for a system and method which enables full replication of server functions at a browser, which may be operating off line from the server, which functions include application code, data, and security.

Previously in web technologies, generation of graphics was done manually. A graphics artist would use Adobe Photoshop or Paintshop Pro, and similar applications, responsive to a specification provided by a requester for a particular graphic, and to generate for it a graphic file, in, for example, a graphics interchange format (.gif). This is a time consuming process, and not dynamic in the sense of generating web sites, etc.. A designer needs to lay out the fonts, graphics, etc, and may need to change the character set to make the resulting graphic viewable in other countries or cultures. Consequently, there is a need in art to get graphic artist out of the loop, and make the process dynamic.

Heretofore, a user could not interactively create and modify a web site. The process for creating a web site has been to use the file transfer protocol (FTP) to upload files that define the site. There is a need in the art for a way to enable a user at a browser to create a site, including a place and rooms, and then to create and modify folders and make other structural changes to the site.

Heretofore, templates and wizards have existed for designing the aesthetic characteristics of a web site. However, there has been no way for the aesthetic characteristics of a parent object to be inherited by its children objects.

Heretofore, people can design forms locally using HTML and upload them to their web site. The problem in doing this is that such a form is strictly a layout of fields, and there is no logic that defines where and how the data should be stored. Consequently, there is a need in the art for a server having facility for accepting and storing form data in a page.

Workflow involves having someone fill out a form and submit it on the web. The form then travels to individuals

in a prescribed set and order. Typically, for providing workflow on the web, a Domino designer designs a data base containing the form that is used by others having access to that database. There is a need to allow users, not skilled as designers, to create workflow documents from a web browser.

It is an object of the invention to provide a system and method which enables a user to accelerate productivity by easily creating and maintaining collaborative spaces on a network.

It is a further object of the invention to provide a system and method which enables a user to customize security attributes of rooms in a collaborative space including a hierarchy of rooms.

It is a further object of the invention to provide a system and method for providing collaborative space for each project of a plurality of individual browser created and managed projects.

It is a further object of the invention to provide a system and method for creating collaborative space that is expressive and offers a range of features that allow

individuals to express their own personalities, such as through the use of graphics and animations.

It is a further object of the invention to provide a system and method for creating collaborative space that is readily accessible to the user, capable of being installed from a web service or from a CD with very little effort.

It is a further object of the invention to provide a teamwork system and method for enabling teams and workers within or between companies to easily set up a common electronic workplace to communicate, share and organize information and documents around any task, project, or initiative.

It is a further object of the invention to provide a team ware system and method which supports small teams that are working together on very project centered or activity oriented things.

It is a further object of the invention to provide a system and method for uploading a document without opening it but by selecting and putting the file to a place automatically.

It is a further object of the invention to provide a system and method for uploading documents which involve the more intuitive process of dragging and dropping the document to a spot on the site where it is to be rendered.

It is a further object of the invention to provide a system and method including within a web site a context for files allowing a user at a browser to create a new page and determine its location.

It is a further object of the invention to provide a system and method for automatically converting a file while dragging the file to a new place.

It is a further object of the invention to provide a system and method for drag and drop uploading from a browser to a server of non-HTML files viewable as HTML files and editable as non-HTML files.

It is a further object of the invention to provide a system and method for managing security of files in a hierarchical database such that security may increase or decrease at a child with respect to its parent.

It is a further object of the invention to provide a

system and method for managing a hierarchical database such that indicia only identifying objects to which the user does have access are displayed.

It is a further object of the invention to provide an end user a dynamic system and method for generating graphical displays.

It is a further object of the invention to provide a system and method for enabling a user to dynamically create and modify a web site.

It is a further object of the invention to provide a system and method for enabling browser creation and modification of web sites.

It is a further object of the invention to provide a system and method for enabling uninterrupted online and offline updating of a web site.

It is a further object of the invention to provide a system and method whereby the aesthetic characteristics of a parent room may be inherited by its children rooms.

It is a further object of the invention to provide an

offline user a full web site interaction experience.

It is a further object of the invention to provide a system and method whereby a user at a browser may create forms, and incorporate them into a room.

It is a further object of the invention to provide a server system and method for accepting and storing form data in a page.

It is a further object of the invention to provide a system and method for enabling a browser to define a work flow document and process.

SUMMARY OF THE INVENTION

In accordance with the invention, and system and method is provided for dynamically generating viewable graphics. A server is operable for storing graphics and text received from a first browser as an image in a cache and parameters for the image in said database. Responsive to a request from a requesting browser an image, for serving the image from the cache and, if not found in the cache, generating

anew the image from the parameters stored in the database.

Other features and advantages of this invention will become apparent from the following detailed description of the presently preferred embodiment of the invention, taken in conjunction with the accompanying drawings.

BRIEF DESCRIPTION OF THE DRAWINGS

Figure 1 is a system diagram of a prior art Internet environment.

Figure 2 is a system diagram of a prior art Notes/Web environment.

Figure 3 is a system diagram of a server having caching capability adaptable for use in the system and method of the invention.

Figure 4 is a system diagram of a prior art computer which may be used at a client or server node in connection with the invention.

Figure 5 is a system diagram illustrating the place server of the preferred embodiment of the invention.

Figure 6 is a system diagram illustrating QuickPlace server databases and templates.

Figure 7 is a diagram illustrating QuickPlace objects.

Figure 8 is a diagram illustrating the creation of a place.

Figure 9 is a system diagram illustrating components of the place server of Figure 5.

Figure 10 is a diagram illustrating a hierarchical structure of rooms with room based security.

Figure 11 is a diagram illustrating a forward pointer in a room from Figure 10.

Figure 12 is a system diagram illustrating file updating.

Figure 13 is a system diagram of the collaboration

system of the invention.

Figure 14 is a system diagram illustrating the QuickPlace server of the preferred embodiment of the invention.

Figure 15 is a diagram illustrating processes executed by the QuickPlace server of Figure 14 for executing browser URL commands with respect to a QuickPlace database.

Figure 16 is a flow diagram of a method for publishing a file on the Web.

Figure 17 illustrates a QuickPlace user room browser display.

Figures 18, 19 and 20 illustrate the presentation of room navigation.

Figure 21 illustrates a graphics server.

Figure 22 is a flow diagram of the method for producing and caching web images.

Figure 23 is a diagram illustrating the insertion of

graphic material into a text document.

Figure 24 is a flow diagram of the method for publishing and caching a document image with graphics.

Figures 25 and 26 are a flow diagram of the method for serving an image from cache or construction.

Figure 27 illustrates a QuickPlace.

Figure 28 illustrates the creation of a subroom.

Figure 29 illustrates the user interface to a QuickPlace room.

Figure 30 illustrates a gallery resource.

Figure 31 illustrates a method for selecting or tweaking a room decoration.

Figure 32 illustrates a method for serving room decoration settings.

Figure 33 is a system diagram illustrating in context the architectural components supporting the replication

function.

Figure 34 illustrates the generation of a document and items from forms and fields.

Figure 35 illustrates the generation of a server prepared document.

BEST MODE FOR CARRYING OUT THE INVENTION

In accordance with a preferred embodiment of the invention, a collaboration system or teamware server platform is provided which is powered by the Lotus Domino system and uses a standard Web browser. A product implementation of this embodiment is referred to as QuickPlace (TM). This browser-based product provides a system environment for collaboration in the virtual world of the Internet characterized by rich text editing and drag-and-drop uploading, server-side graphics rendering, scene-based templates (or, wizards) for performing administrative tasks, and hierarchical security.

In accordance with the invention, a server system is provided which enables users to create space on a network that is easy to create, making them productive right away. All that is required to create that space is a name, a place name, and a password for access. The QuickPlace approach is to give a sense of space where information can be entered and posted, where people go to collect thoughts and collaborate, and to provide a place or space characterized by ease of creation and maintenance. The QuickPlace approach provides a place for each project.

Exemplary Server/Client System

By way of background, Figure 4 illustrates the system architecture for an exemplary server 105 or client computer 101, such as an IBM THINKPAD 701® computer or like computer, on which a network access system can be implemented. The exemplary computer system of Figure 4 is discussed only for descriptive purposes, however, and should not be considered a limitation of the invention. Although the description below may refer to terms commonly used in describing particular computer systems, the described concepts apply equally to other computer systems, including systems having architectures that are dissimilar to that shown in Figure 4.

The server 105 or client 101 computer includes a central processing unit (CPU) 116, which may include a conventional microprocessor, random access memory (RAM) 117 for temporary storage of information, and read only memory (ROM) 118 for permanent storage of information. A memory controller 113 is provided for controlling system RAM 117. A bus controller 114 is provided for controlling bus 111, and an interrupt controller 115 is used for receiving and processing various interrupt signals from the other system components.

A diskette, CD-ROM, or hard disk may provide mass storage. Data and software may be exchanged between client 101 and server 105 via removable media, such as a diskette or CD-ROM. A diskette is inserted into diskette drive 123, which is connected to bus 111 by controller 122. Similarly, a CD-ROM can be inserted into CD-ROM drive 125, which is connected to bus 111 by controller 124. A CD-ROM can also have digital versatile disc (DVD) playback capabilities as well. Finally, the hard disk is part of a fixed disk drive 127, which is connected to bus 111 by controller 126.

User input to the server computer 105 may be provided by a number of devices. For example, a keyboard and a mouse

may be connected to bus 111 by keyboard and mouse controller 120. An audio transducer, which may act as both a microphone and a speaker, is connected to bus 111 by audio controller 112. It should be apparent to those reasonably skilled in the art that other input devices, such as a pen and/or tablet and a microphone for voice input, may be connected to server computer 105 through bus 111 and an appropriate controller. DMA controller 119 is provided for performing direct memory access to system RAM 117. A visual display is generated by a video controller 121, which controls video display 129.

Server computer 105 also includes a network adapter 128 that allows the server computer 105 to be interconnected to a network 110. The network 110, which may be a local area network (LAN), a wide area network (WAN), or the Internet, may utilize general-purpose communication lines that interconnect a plurality of network devices.

The Web server 105 answers URL (Universal Resource Locator) requests by sending back pages of data encoded in HyperText Markup Language (HTML). It also handles URL requests and HTML forms that trigger executable programs according to the Common Gateway Interface (CGI) specification. The Web server 105 includes code that manages

both inbound and outbound HTTP (HyperText Transfer Protocol) communications. In these respects, the Web server 105 performs like any other HTTP server, responding in the standard way to standard URL requests. The preferred embodiment will be discussed primarily in terms of a Lotus Domino web server 105 although the system and method of the present invention may be implemented in any web server.

By way of further background, as can be seen in its most basic form in Figure 2, web server/client system 130 and Notes network 131 include a Domino web server 132 having many tasks running on it simultaneously. Among the server tasks are the Domino database server tasks 137, i.e., serving up documents from Domino databases 142, and the HTTP server tasks 134, i.e., serving up documents having formats such as HTML, GIF, JPEG, XML, DHTML, BMP, MPEG, WAV, Java applets, and other file formats known to those skilled in the art from file system 139 via common gateway interface (CGI) server 133.

Notes software, available from Lotus Development Corporation, works with Domino to provide a distributed client/server database application to let users organize, process, track, and use information to suit their individual needs. Notes/Domino consolidate the tools needed to

effectively communicate and collaborate in an organization by providing, inter alia, email, group discussion, workflow, scheduling, document management and many other functions. Domino databases are built on three basic concepts: documents, views and forms. Documents are collections of data items which can be retrieved as a set. Views are the ways of accessing the indices or summaries of documents stored in a database while forms are templates for accessing and displaying documents.

When a Notes client 141 requests access to a Domino database 142 via the Notes network 140, the Domino database server task 137 provides access. When a web client 101 requests an HTML document, the HTTP server task 134 provides it. When a web client 101 requests a Notes document, the HTTP server task 134 passes the request through to the Domino database server task 137. If access is granted, the Domino database server 137 retrieves the requested document and passes it to an HTML converter 136 which converts the Notes views, documents, and forms from Notes format to HTML format, then delivers the resulting HTML pages to the HTTP server 134 for serving to the web client. If a web client submits a form or query, the HTTP server task 134 passes the form to the HTML Converter 136 which converts the form to Notes format and passes it to the Domino database server 137

for appropriate processing.

Heretofore, to set up a web site, connection 109 between HTTP server 134 and network 100 is disconnected while an HTML programmer 138 modifies file system 139. In accordance with the present invention, such a disconnection is not necessary.

Referring further to Figure 2, in operation, HTTP server 134 takes URLs from a client 101, and looks to file system 139 to get the URL to serve to client 101. Domino database 142 provides Notes functionality (workflow, security, etc.) to a web server 132. A web server includes HTTP server 132 and file system 139, CGI server 133 which provides a parallel path HTTP server 134 to file system 139.

QuickPlace objects are implemented on top of Notes.

Notes server 137 provides an interface for enabling an owner to provide Notes functions to web servers/clients. Server 137 enables a user to use Notes functions via a web client browser 101 on the Internet 100. This it does by providing in HTTP server 134 an HTTP to Notes converter 135. Thus, a command or URL from web client 101 in HTTP protocol (get, getnext, openform, getmail, etc.) is parsed as a Notes

command in converter 136 and sent to data base server 137. Buried in the URL is the Notes command that is parsed out.

To access E-mail, the command to do so is included in the URL command, this is parsed out and sent out to database server 137, which accesses database 142. HTML converter 136 converts .nsf (notes store facility) to HTML format, which it then provides to HTTP server 206, to be served to web client 101.

In accordance with a preferred embodiment of the invention, a QuickPlace server is a modified Dominos server 132. QuickPlace is a virtual server on the same machine as, for example, many other domino and/or QuickPlace servers.

Referring further to Figure 2, QuickPlace is a smart master (also referred to as a wizard), a template for creating a web site which doesn't require an HTML programmer 138 to set it up. Previously, a programmer 138 would have to program into the file system 139 the code needed to set up the web server Internet site with the web server disconnected at 109 from the web site. In accordance with the present invention, web client 101 or notes client 141 can design site without bringing down the web connection

109. This is done by starting with a default web site and build from there. The default is built upon by using point/click, color selection, room creation rooms, file drag and drop files, input files directly to site, control access to site (manager, author, reader), and so forth.

Web Server Cache

Figure 3 is a block diagram of a web server system having a caching system useful in connection with the preferred embodiments of the present invention, and is an example of a domino server without QuickPlace shown. As will be described more fully hereafter, QuickPlace allows a user at browser 101 to use a default website and from that as a manager to set security, pop up a window, add reader/author, distribute notifications and passwords to control access, create a room which gives options for customizing color scheme, control access to the room, create sub room down as many layers down, with each subroom selectively inheriting characteristics of a previous room, and changing these characteristics.

As shown in Figure 3, the web server 132 may be connected to a number of Domino sources 142. However, the

sources may comprise any number of different types of elements, other than Domino databases: other databases, files, other web sites, etc, but Domino sources are shown for clarity. The web server may also be connected to HTML databases 139 as was discussed above. The web server 132 comprises and is associated with many functional units. It comprises the HTTP server 134, discussed above, which comprises a TCP/IP application 157, and a HyperText Transfer Protocol (HTTP) unit 156. The web server 132 further comprises the HTML converter unit 136 discussed above. It further comprises a parser 145 (for parsing received URLs), a cache 143, cache control unit 144, and operates in cooperation with a cached-response analyzer 146, and a database server 137, including a response builder 154, a source access unit 155, and a cacheability analyzer 149.

These units operate as follows: TCP/IP unit 157 and HTTP unit 156 act together as the interface to the Internet by implementing the TCP/IP and HTTP protocols for server 132. TCP/IP unit 157 utilizes the TCP/IP protocol for conveying and receiving information to and from the Internet. HTTP unit 156 implements HTTP, which is the standard on which the Web operates. These two units provide the full-service interface to the Web.

When server 132 receives a URL from a client, the HTTP server 134 passes the URL to the URL Parser 145, which breaks the URL into different parts. The parsed URL is passed to the cache control unit 144. With a Domino server, within the URL that is received from the requesting user is a Domino/Notes-specific command, which indicates what action is being requested. The following are examples of server specific commands within the URL that may be received by the Domino server:

?OpenDatabase - command for opening a database;

?OpenView - command for opening a view;

?OpenDocument - command for opening a document;

?OpenForm - command for opening a form;

?ReadForm - command for reading a form; and

?EditDocument - command for editing a document.

While, in this example, each of these commands has a "?" in front of the command as syntax that the server can use to identify the string as a command, the server can identify other syntaxes as well. These commands require a response to be sent to the requesting user. The requested response may have already been cached and it may be valid and applicable. For those URLs having commands requesting a possibly-cached response (i.e., ?OpenDatabase, ?OpenView,

?OpenDocument, ?OpenForm, and ?ReadForm), the cache control 144 examines the request against previously cached responses to determine whether any of the previously cached responses is appropriate for the request. This is done by comparing the parsed URL against the URLs of the previously cached responses in the cache 143. If there is not an exact match or if the URL doesn't have "cacheable" commands (e.g., ?EditDocument), the parsed URL is passed to the response builder 154. The response builder 154 uses the parsed URL to build the response by accessing the appropriate sources (via source access unit 155) and retrieving the appropriate "parts " to construct the response. The parts retrieved by the response builder 154 may comprise many different types, including data, forms, subforms, database design elements, calculations, etc. In other words, there is no theoretical restriction as to the type of parts comprising a web page response. These parts each have their own attributes. For instance, some parts may or may not have last modified dates associated with the part. The attributes of all of the parts used to build the response are collected and analyzed by attribute analyzer 153. The attribute analyzer 153 builds a "composite" of the attributes, the attribute composite being representative of the entire response.

Once the web page response is built by the response

builder 154, it is passed to the HTML unit 136 for conversion to HTML. This HTML response is then passed to the HTTP server 134 for serving to the requesting user. At the same time, the attribute analyzer 153 passes the composite of the parts' attributes to the cacheability analyzer 149 for determining the cacheability of the built response. The cacheability analyzer 149 examines the attribute composite and, if it determines that the response cannot be cached, the response is not cached. If it determines that the response can be cached, it provides an indication to the cache control unit 144, along with the response and an associated set of cache strategy indicators generated by the cacheability analyzer 149. These indicators are used by the cached-response analyzer as discussed below. The cacheability analyzer 149 comprises a cacheability analyzer interface 150 and a caching strategy generator 151. The cacheability analyzer interface 150 acts as an interface for the cacheability analyzer 149 while the caching strategy generator 151 examines the attribute composite and creates a caching strategy.

If the cache control unit 144 determines that there is an exact match between the parsed URL of the user request and the URLs corresponding to one of the cached responses in the cache 143, the candidate cached response along with its

associated cache strategy indicators is passed to the cached response analyzer 146. The response analyzer 146 performs two series of tests. The first series of tests are response-specific and the second series of tests are request-specific. The response-specific tests are performed by the validity analyzer portion 148 while the request-specific tests are performed by the applicability analyzer portion 147. If the candidate cached response and its associated attributes pass the two tests, the candidate cached response is simply served up to the user at browser 101 via the HTTP server 134.

Co-pending patent application S/N _____, filed _____ assignee docket LO998030, provides additional detail with respect to operation of servers 132, 137 and related components.

Collaboration Space Server

Referring to Figure 5, the Domino structure 132 is enhanced with QuickPlace extensions 160 to provide a Notes/web interface for providing collaboration. In a preferred embodiment of the invention, a full blown Domino server is installed, including web server 134 and Notes

server 137, and then the QuickPlace module 160 is installed as an overlay.

Referring to Figure 9, in another preferred embodiment a QuickPlace server 163 includes web server 132, domino web interface 134, a QuickPlace module 160, and a notes server, providing a product to which web browser 101 and Notes clients can connect. Objects within QuickPlace module 160 include a place with rooms, rooms having folders (optional) and pages; and a place has members. When a place is created, one room is created in which pages can be created. A place is created from a placetype, which is a template that specifies the number of rooms, members, security for rooms, look and feel, and forms and folders in each room. A place type was once a place, which is saved as a template from which new places can be created. A place is a directory which is created from a template by entering place name, person name and desired pass word. The directory is created in file system 139, with one database for each room. Thus, a room is a QuickPlace object that maps to a Domino database. The manner in which this mapping occurs will be described hereafter in connection with Figure 8.

Referring to Figure 14, a system architecture view of the QuickPlace server 137 is presented in the context of a

system including a web browser 101 and Domino server 132. Domino server 132 includes HTTP server 134, URL processor 188, and Domino engine 189. QuickPlace server 137 includes URL processor 164, open document handler 165, edit document handler 166, open form handler 167, form/scene loader 168, command handler 169, formula handler 178 and HTML generator 179.

URL requests are received by Domino 132 from browser 101, and are of the following syntax:

(1) http://server/dir/db /view|folder|form /doc?command &command arguments

Examples of URL requests are:

(2) http://QuickPlace/ACME/Main.nsf/PageUI?OpenForm

(3) http://QuickPlace/ACME/Main.nsf/Discussion/ docID?EditDocument&Editscreen=" "

A URL request received from browser 101 is fed by HTTP server 134 to URL processor 188, which examines the URL to determine if QuickPlace is specified as the server. If not, the URL is passed to the domino engine for process. If

QuickPlace is specified, the URL is passed to QuickPlace server 137 where it is received by URL processor 164.

URL processor 164 examines the URL from the browser, received via Domino server 132, to determine the command, and depending upon that command passes processing to open document handler 165, edit document handler 166, or open form handler 167. In resolving these commands, handlers 165-167 reference a library of Domino and QuickPlace classes, including the following:

Domino classes:

NDatabase
NView
NNote
NDocNote
NFolder
NForm
...

QuickPlace classes:

HuUiDoc
HuForm

HuView

HuDocNote

...

Responsive to an open document command in a URL including QuickPlace in the URL prefix, open document handler 165 instantiates a database class using the Domino and QuickPlace classes, which involves opening the database containing the document, the view containing the document, and the document itself. This involves a call to form/scene loader 168, which determines and loads the proper form and the scene. A form describes the document items of interest to a particular request, and the layout of the page for that request, as specified by the URL from the user browser 101. In the process of loading the form and scene, loader 168 may determine that there is a formula; and if so, it is passed to formula handler 178 for processing. HTML generator 179, responsive to loader 168 and, if a formula is included, to formula handler 178 to generate the HTML and send it to the user browser 101 via Domino 132.

Responsive to an open form command in a URL including QuickPlace in the URL prefix, open form handler 167 is called by URL processor 164 to create a new document in the database specified by the URL, inside the view or folder

specified in the URL. Open form handler 167 calls form/scene loader 168, which loads the appropriate form and scene. This may require a call to formula handler 178 to resolve any formulas in the form, and HTML generator 179 takes the information provided by form/scene loader 168 and, possibly, formula handler 178, and passes back to Domino 132 the HTML for display at web browser 101. If there are any commands to be executed as a result of creating a document, open form handler invokes command handler 169 to execute the command. These commands includes create, delete or edit room, user, or folder; or the publishing commands make draft, make draft from published, publish, save under construction, revert, and discard. The make draft command is invoked to create a document. The publish command publishes the draft which then becomes published (a document in which a published field gets set to on). To change a published document, the user issues a URL including the edit document command. QuickPlace server 137 creates a draft document from the published version. The user can change the draft document, and then issue a command which either reverts the draft document back to the published version, save the draft as a document under construction, or discard the draft.

Thus, in overview, the open form handler 167 operates

according to the following procedure.

- Browser 101 issues a URL containing the open form command, something like the following:

  URL:/QuickPlace/ACME/Main.nsf/PageUI?OpenForm

- Domino sees that the server specified in the URL is QuickPlace, and passes the URL to QuickPlace 137.

- URL processor 164 parses the URL and determines that it is an OpenForm command, and passes it to open form handler 167 for processing.

- Open form handler 167 determines that the command is for the ACME place, for its top room Main.nsf, and using the pageUI form.

- With that information, open form handler 167 creates a draft document by calling the make draft publishing command. (When a room is created, first a draft is created, which is then published, and then the create room command is executed.)

- So, open form handler 167 has first created a draft.

With that it calls form/scene loader 168 with the given form and scene for the draft document, which it loads in memory. As part of loading the draft document, formulas may be computed in formula handler 178 - and this may involve the selection of default values. Once the draft document is loaded in memory with the given load and scene, form/scene loader calls HTML generator to generate the HTML to be sent to browser 101 via Domino 132, and the document with its fields is presented to the user.

Responsive to an edit document command in a URL including QuickPlace in the URL prefix, edit document handler 166 executes the procedure to be described with reference to Figure 15. This command is similar to open document, except in the case of edit document the document already exists in database 230 for the QuickPlace specified in the URL from browser 101. In this case, suppose the URL is of the following form:

url: /quickplace/acme/main.nsf /discussion /docid?editdocument &editscene

where acme is the directory, main.nsf is the database, discussion is the folder, docid is the document,

editdocument is the command to be executed, and editscene is the first of a list of command arguments – a list of arguments, each argument separated by the & delimiter. Execution of this command will result in opening given document docid, which is located in acme place, top level room main.nsf discussion folder. To do so, edit document handler 166 uses the Domino and QuickPlace classes to open folder discussion and document docid, and calls form/scene loader 168 to load appropriate form and scene from haikucommonforms database 236 (forms is older terminology for scenes).

A scene controls the layout and defines the data to be captured from user, such as type of QP to be created. A scene also specifies the actions or choices to be presented to the user, such as next or back. Next brings up another scene, and a series of scenes can be used, for example, to change the type of a QP place (place type). Thus, scenes are like steps, each scene is represented by a screen for display at browser 101, and the user can move back and forth. This will be described, for the case of the edit document command, with reference to Figure 15, which illustrates the edit cycle executed under control of edit document handler 166.

Referring to Figure 15, in connection with Figure 14 and the URL command issued from web browser 101 by the user of:

url: /quickplace/acme/main.nsf /discussion /docid ?editdocument &editscene

who desires to edit the document docid in database 230. The process involves the creation and use of scenes 224, 226, 227 and 228. Scene 224 has been created previously in response to an open document command from browser 101, as explained above, which caused form/scene loader 168 to go to QP database 230 to retrieve document 223 from which scene 224 was derived and displayed as a screen at browser 101. Scene 224 gave the user the opportunity to select NEW, resulting in an open form URL request being sent by browser 101 to server 137, which resulted in open form handler 167 creating a folder document 225 in memory and in forms/scene loader selecting and returning to browser 101 display scene 226. Scene 226 gives the user the opportunity to select an edit command, and the above URL1 resulted from that selection.

Recapitulating, open form created document 223, selected the scene A 224, generated corresponding HTML1 and

sent it to browser 101. Now browser 101 displays document 223 (which is created at the browser by scene 224), and by the user selecting NEW, open form is issued to the server, which will open form and return scene B 226. User is now in the edit cycle, in the process of supplying information to the server 137 through the scenes 226 and 227 in order to created the document 225 - it is a multi-step process. It is a cycle because it a has a beginning and an end, and is edit because document 225 is being modified.

Thus, with browser 101 displaying scene B 226, the user is given several choices to perform actions (all scenes have actions, represented by buttons, menu choices, or the like) and, in this case, NEXT is an action option. When NEXT is selected, an edit document request URL, given above, is sent from browser 101 (because, in this example, NEXT refers to a previously created document). In response, edit document handler 166 opens document 225, form/scene loader selects scene C 227, HTML generator 179 generates html and sends it back to the browser 101 and scene 227 is there displayed.

Now browser is displaying scene C 227. The user enters information, such as name of document, text, etc. Scene C 227 presents to the user, for example, the actions of: publish, publish as, save, and cancel. Assuming the user

selects publish, a scene is displayed asking for more information, such as where to put page, or the name of the room being created.

Handlers 165, 166, and 167 having now been described, and publishing methods, commands will be further explained. Commands get invoked for creating, deleting or changing QuickPlace objects. These commands include, as enumerated above, create, delete, update place, room, user or folder. Other commands include providing for changing security in a room.

Continuing with reference to Figure 15, one cycle of command invocation, for create folder, is as follows:

- user clicks on new when browser 101 is in scene A 224
- new issues open form to bring scene B 226 to browser 101
- user chooses create folder
- user select next
- QuickPlace 137 creates an empty folder document 225
- server 137 returns scene C 227 to browser 101
- which lets user choose the kind of folder to be created (various options are presented)
- user selects folder type=standard list, and clicks on

next

- browser sends choice to folder type = standard list as a command argument, to be stored in a field in new document 225
- server 137 sends back a next scene (not shown), asking for name of folder
- user enters "draft letter" for name of folder
- user select next to submit changes,
- another scene is returned asking where
- user answers, and selects next
- animation is displayed at browser 101, and as browser is in transition from this last scene, a create folder command 229 is executed to create and store in database 230 the actual folder object, that uses characteristics specified in new document 225.

Alternatively, the user could have selected other options which would result in execution of other commands 229, such as create user, and so forth.

Room Security

Referring to Figure 10, QuickPlace rooms 201-204 and 210 are connected by forward and backward pointers 205-209

and 211, and these enable the security of each room to be independently managed. Each room has its own security; that is, the identity of each user allowed to enter the room and that users security level: the three levels being reader, author, manager. This is held in an access control list which is a part of each room. While an individual, say Steve, has reader access (R) to the library 204, he can have author (A) access to a subroom 211. This enables a subroom 211 to have increased/ maintained, or decreased access authority for a particular individual with respect to its parent room 204. Only individuals with access to a parent 204 can access a subroom 210, but that subroom 210 can have changed access for the subroom 210 for these individuals. Previously, security could not be increased in subrooms 210 with respect to a parent room 204.

A database access control list (ACL) specifies who can or cannot access the database. For users who can access a database, access levels and roles determine the specific actions they can perform -- for example, creating or deleting documents. Document access fields (Readers and Authors fields), in conjunction with the database ACL, control who can read or modify specific documents. Thus, to limit access to specific documents created from a form, a readers field is included. A readers field explicitly lists

the users who can read documents created from the form. If a form has an access list, names from the readers field are added to the form access list. Otherwise, the readers field controls access to documents created from the form. Entries in a readers field cannot give a user more access than what is specified in the database access control list (ACL); they can only further restrict access. An authors field works in conjunction with author access in the database ACL. Listing users in an authors field expands access rights by allowing listed users to edit documents they create. Entries in an authors field cannot override the database access control list; they can only refine it. Authors fields affect only users who have author access to the database.

Referring to Figure 11, forward pointers 205, 209 are secure. Security, in this context provides that forward pointer 205 to project A 203 carries the same security as that of project A 203, and anyone viewing main room 201 who is not entitled to access project A 203 will not see room 203 listed in parent room. QuickPlace does not show a user things or objects to which the user does not have access. In past, such objects were shown, but were greyed out or otherwise managed so that user access was inhibited. Forward pointers, therefore, include room name field 212, address to database name field 213, and readers field 214,

which includes a table of user identifiers 215 for each user permitted to access the room, with corresponding access authority 216 for each such user, which may be manager, author, or reader.

Room Modification from Browser

Referring to Figure 12 if a user client 101 wants to modify the QuickPlace database 142 file 221, then QP server 163 creates a new qpfile' 222 for client 101 to change without bothering user client 102 who may still be accessing the qpfile 221. Once done, qpfile' 222 replaces qpfile 221.

Referring to Figure 13, a qpfilex 235 resides in database 142. A browser on client computer 101 accesses a file system 232 on the PC which is a replica 234 of qpfilex within a replica 233 of Domino web server 163 (or at least a portion of it) on it as well. In this manner, client 101 has the necessary function locally to do off line browsing, changing qpfilex 235 with new data, new rooms, etc. to create qpfilex' 234. When done, client 101 with network 100 access replicates file qpfilex' 234 to the main site to replace qpfilex 235 with qpfilex' 234.

Referring to Figure 6, the QuickPlace system includes browser 101, QuickPlace databases 161 and QuickPlace templates 162. Mediating the conversation between the browser and the databases and templates is a Domino web server 132 with QuickPlace extensions 160. Templates 162 contain HTML and Java Script and controls and page generation language. Databases 161 stores common resources, sign elements, and client components including rich text aesthetic control, and applet control which enables server and client desktop integration and offline control. With respect to offline control, with a QuickPlace created on a server, the offline feature allows replication, or installation of that place on a user's local computer, the purpose of which is to allow the user to work disconnected. Sometimes, disconnected work will be faster than accessing the server, depending on the network bandwidth.

Object Model

Referring to Figure 7, the QuickPlace object model 170 is described, together with the Domino equivalent 180. Place 171 consists of rooms 172. A room 172 is made up of pages 174. Folders 173 are used to organize pages. Members 175 belong to rooms, and are those users authorized to

access them. Place type 176 controls the creation of a place, including how many rooms it has, for example. Room type 177 controls the appearance and content of rooms. This simple object model is implemented using equivalent Domino objects 180. A place 171 is the QuickPlace site, organizes pages in rooms and folders; it is a simple file directory 181 containing rooms 172. A folder is an organizing structure for collecting and displaying related pages in a site. Each room 172 is a database 182, a collection of pages with discrete security and authentication. This enables access control properties to be carried from the database 182 over to the room 172, enabling places 171 on a database to grow. QuickPlace folders 173 are implemented as folders 183, and sometimes as views in the background for collecting documents and for saving a stored query. A QuickPlace page 174 may be interpreted as a data note 184, the combination of a data note, a form, and one or more subforms, and is also referred to as a document. Subforms control the way that the page is viewed and the user interface. A form manages the display of data notes. A form can contain fields for containing data and employ scripts to process and compute data. A page is the basic vehicle for content. Content is created using the QuickPlace editor or importing content from an external source. A member 175 is also a data note, and each place

contains its own member directory. A member note contains information about a tem member of a QuickPlace. In addition to this data, the member must be listed in the ACL of main.nsf and names.nsf to pass authentication. Place type 176 and room type 177 are implemented as Notes templates 186, 187, respectively. A place type 176 provides the structure and design for the resulting place. A room type 187 provides the structure and design for the resulting room.

A database 182 is a container for data, logic, and design elements in an application. Design elements are building blocks used to create an application, and include pages, forms, outlines, navigators, views, folders, framesets, shared resources and agents. Domino databases have the file extension .nsf.

A folder 183 is a container used to store related pages (documents) or groupings of pages (documents). Herein, pages and documents sometimes used as equivalent, and in other contexts a document is created by a server from a page submitted from a browser. Folders have the same design elements as views, and are created in much the same way as views, using the Create-Design-Folder command. The difference between folders and views is that views always

have a document selection formula that collects and displays documents automatically. A folder remains empty until users or programs add documents to the folder.

A form, like a page, displays information, but unlike a page is also used to collect information. A form provides the structure for creating and displaying documents, and documents are the design element that store data in a database. When a user fills out the information in a form and saves it, the information is saved in a document. When a user opens the document, the document uses the form as a template to provide the structure for displaying the data.

A template is a file that contains the structure for a database - that is, pages, forms, and views - but does not contain documents. It is a skeleton that contains design elements, but no documents. Documents have .ntf as their file extension.

Place Creation and Modification

Referring to Figure 8, a place 191 is created. A place type 190 in its basic form is two databases, one that describes the room type 192, in this case, a meeting room;

and the other that describes the member directory 194 for that place. A place 191 is created by copying both templates 192, 194 into a database. The top level room 193 of every place is called Main.nsf for ease of access and contact.nsf is the member directory for that given place.

A QuickPlace is created by creating two databases from database templates. The Main.nsf is created from a MeetingRoom.ntf template, and is the parent database in a QuickPlace - all other databases in a QuickPlace are children of Main.nsf. Contacts1.nsf is created from the Contacts.ntf template.

A room is created from a default room type template, PageLibrary.ntf, which provides indexing infrastructure for maintaining the pages in a room, and also security and authentication features so that access to a room can be limited to a subset of team members. The database created from the PageLibrary template is assigned a unique name by the system to allow for multiple rooms within a QuickPlace.

When a QuickPlace server is installed on a computer, a QuickPlace directory is created under the domino server data directory. This QuickPlace directory includes several objects of interest including an Error types subdirectory

and a page library subroom with unique name. Names for these objects are created to be unique because of the offline feature. Many users could be creating rooms, not on the server, on their own computers. And when these replicate off of the server, they could collide by name unless some uniqueness is guaranteed.

Server Extensions

Referring again to Figure 6, eight QuickPlace extensions 160 are enhancements made to the Domino web server 132 in order to support a QuickPlace application. These extensions 160 are enabled only for QuickPlace URLS; that is, they are enabled for URLs that are targeted against a particular QuickPlace. These extensions are: (1) shared design elements, (2) database linkage, (3) commands, (4) publish and draft model, (5) security and authentication, (6) forms and fields, (7) decoration model), and (8) graphics server.

(1) Shared design elements are shared forms stored in a common template. These elements are provided to deal with the situation where databases are created with duplicate forms and subforms. To avoid having all of many forms

would be duplicated with every room, they are gathered together in a common template, referred to as a head group common forms. As a result, databases are smaller since they don't have all these assigned elements. And because they're smaller, they can be retrieved and served faster and enable efficient server site caching.

(2) Database linkage enables the grouping of a number of databases in a hierarchical way. A place is a collection of databases, and these need to be represented in a parent child relationship. Data notes represent the hierarchy to the database. There is a data note in the parent database, and there is a data note in the child database. The use of data notes for these QuickPlace extensions as a way of representing their functionality has the benefit that there are many ways of manipulating them, whether it's with Java or forms or the Notes designer.

(3) Commands are representations of the methods that operate on QuickPlace objects 170. They can create, modify and destroy them. Examples are create place, create room, create user, modify user, and so forth. When a user interacts with a user interface (browser), he submit a form to the server. When a form is submitted, the server detects a named command item in that form, extracts from the form

any additional arguments, and then performs the command, such as, for example, create a user in the particular member directory. In this way, QuickPlace functionality is encapsulated into a well-defined set of commands.

QuickPlace commands are set forth in Table 1, and the fields of QuickPlace objects are set forth in Table 2.

(4) The Publish and draft model extension allows management of two versions of a page: published and draft. Each time a copy of a published page is to be edited, a draft copy is made and used. A link between the draft and the published versions is achieved by having the draft point to the published copy and the published copy point to the draft. When a draft copy is made, an author field is provided in the published copy to prevent other people from editing what is currently undergoing revision. The reader field in the draft copy is dropped, thus preventing others from seeing the draft copy revisions before it is published. When the draft is ready to be published, the information from the draft is put it over into the publish copy and the draft discarded. The benefits of this extension is a user may enjoy private revision of pages at his site, his places, independent without being visited by other users. While a user is working on a private copy of a page, the public, or

published, version is made available to anybody that visits. This is also the basis for check out functionality, by which a user working on a document has it checked out, preventing others from working on it. This is also the basis for work flow: as page drafts move between reviewers, the readers and authors fields on the draft are manipulated to selectively enable and inhibit access or modification. Once a draft page is approved, then it becomes published.

(5) The security and authentication QuickPlace extension is consistent with the QuickPlace model, which provides three levels of security or roles: reader, author, and manager. There exists a member directory for each place. What this means is that each place has its own set of members that visit it. The Domino server is modified to perform local authentication against that directory, making places very portable, self-contained. And they don't collide with other members in other places. A user, having control of his own place member directory, set his own security for access to that directory.

(6) The forms and fields QuickPlace extensions control the content and formatting of pages. They're very fundamental. They're also implemented as data notes, and because they're data notes, they allow for easy customization.

(7) The decoration model extension allows user control over the customization of rooms and places. Place and room properties and aesthetics including look and feel are stored in notes, main.nsf. Also, there is a shared graphics resource database that controls all the graphics that are shared by all of the rooms. As a consequence data notes and databases are easily extended and customizable.

(8) The graphic service extension transforms text, font, and effects into an image through server side rendering. This us URL driven, and allows rich text formatting independent of browser capabilities. That is, regardless of the capabilities of the client, because text, font and effects are rendered at the server, universally acceptable jiff may be served. This is done through the command Get Image, which includes arguments specifying the text that to be transformed, and the desired colors, font name, sizes, and animation (that is, undulation), shadow color, and so forth.

File Upload by Drag and Drop

Referring to Figure 16, the method of the invention for dragging and dropping files is shown. Responsive to a user

dragging and dropping one or more files, a client or a server is operable for converting the files to HTML, detecting support files by parsing the HTML, uploading all of the files to a web server, and modifying the HTML files to display the content of the files in a web context, thereby enabling the author or other users to modify the original files. This process is accomplished without having to open file, and requires no manual conversion to HTML. User is in a desired, familiar place, site, or context where he doesn't need to figure out file names. We provide speed, ease of use. Dragging and dropping a non HTML file into a browser which is presented to users as HTML and presented to others for editing as non-HTML is fundamentally easier than a process requiring the two steps of the user of opening a file application and saving the file as HTML. Also, with QuickPlace, the user drags the document to be copied into the spot in the site where it is to be rendered. In other systems, a user has to deal with directories and file trees which may not be so intuitive for the user as to where the document should go.

In accordance with the invention uploading/downloading drag/drop of MS Word documents, Excel documents, and Power Point presentations, and the like, is accomplished in a way that the data is simultaneously presented as HTML to users

yet preserved in its original file format for future editing through a web browser. A file is taken by a browser and dragged into an area on a form; readers will see an HTML version of that file yet the author, when editing, will still have word file to edit. Author sees the file in its original syntax, every else sees it as HTML, and this through a browser. In Power Point, the user is provided an interface allowing HTML reader to cycle through the pages (of a slide show). To do this, HTML and HTML control for toggling are generated. Both the file and its navigation controls are converted to HTML. In addition, all references to images are found, converted and uploaded automatically. To show a document including a picture to others on the web, text is converted to HTML, and the image is converted into a .gif file. Both files are then uploaded to the server, which makes sure that the HTML that URL references the image points to the correct place in the new storage location at the server. The user performs the steps of creating the page, dragging and dropping the page, and publishing it. The system (client and server) automatically performs the remaining steps.

Referring to Figure 16, in step 241 the user creates the files and in step 242 opens QuickPlace in a browser. In step 243 the user creates a page in QuickPlace (either new

or imported) and in step 244 drags and drops the file or uses the file-open-dialog command to select the file. This ends the user interaction, and the system takes over. In step 245 the system converts the file to HTML if required, and in step 246 parses the HTML to find, in step 247, if any images or other support files are present. In step 248 the HTML is modified for web context, and in step 249 the original and converted files are uploaded to the server. The user may, in step 250, select publish to choose the location (room) within QuickPlace for publishing the file.

Room Navigation

Referring to Figure 17, a QuickPlace user room display includes side bar 263, page 261, and action bar 262. To visually indicate movement of page 261 from room to room, the material in side bar 264 slides in the direction of right arrow 264 to enter a room or left arrow 265 to leave it. Referring to Figures 18, 19 and 20, material 266 enters side bar from the left to show a new room display down the hierarchy is being entered and displayed in page 261; and material 267 moves to the left to enter side bar 263 as material 266 moves out to the left, indicating that the material in display 261 is changing to a room higher in the

hierarchy, that is that display 261 is leaving a room. In this manner, the display projects to the user where he is at any particular time. To show leaving a room, the visual effect is to have the side bar move to the left while the side bar for the room being entered appears from the right.

Graphics and Aesthetics

Referring to Figure 21, QuickPlace server 163 includes a graphics server 350 connected to database 161.

Referring to Figure 22, aesthetic support for a web site, such as QuickPlace, is provided as follows. In step 351, when a user selects QuickPlace, he also selects a theme, which includes set of aesthetic parameters for color, image, fonts type, and other effects. In step 352, the user creates documents which inherit the aesthetic parameters originally selected for this QuickPlace. If at some later time, the user changes the QuickPlace aesthetic parameters, these are propagated to all subsequently created documents. In step 353, the user enters text data for this document, and in step 354 selects 'save'. In response, in step 355 graphics server 350 saves the parameters and text as an image, in step 356 stores the parameters for this image in

database 161, and stores the image itself in cache 153 (Figure 3). This is managed by the graphics server 350. This process is particularly useful in an Internet environment, for people who access the QuickPlace can request via an URL parameter string a particular graphic which will be served to them by the graphics server. A graphics server generated image request is used by any user to make an image, and the graphics server responds by checking the cache and if it is found there, serves a saved image from the cache, and if not, generates it anew.

Referring to Figure 23, a series of panels 360-362 is illustrated showing how a document is published so that it appears as an item 363 in side bar 263 of a QuickPlace display 360. The user selects layout 369 and enters the title in work area 261. The QuickPlace program graphic server 350 puts the title in side bar 263 as a graphic 363, to compliment the aesthetic parameters of the QuickPlace. The user then enters text and selects appearance parameters in work area 261, and these are also generated as graphics which will be displayed in the read area 261 of the saved document.

Referring to Figure 24, the process for generating a document including inserted graphics is as follows. In step

370, the user is served a document on edit screen 361. In step 371, with the cursor positioned at the point 364 in edit screen 361 where the graphic is to be inserted, the user selects insert graphic text button 367. In step 373, insert graphics screen 362 is presented to the user which includes a text area 261, and buttons and drop down menus for selecting various parameters, including effect, animation, color, size, and so forth. In step 374, when the user selects done 365, edit screen 361 is presented with the image appearing at the selected point 364. In step 375, the user selects publish 368. In response, the editor, which has the URL of the insert graphic text screen 361 that was used to generate the desired graphic, issues a request to graphic server 350 for that URL. The graphic server, in step 376 and responsive to request from the editor, either retrieves the previously generated image of the requested URL, or generates the graphic and serves it to the screen 360 in publish mode.

In steps 377 and 378, the image created is not saved in the document, but rather in the graphics server URL that generates the image. This is important: an image that is 1x2 inches may be .5 megabytes, and an URL is much less. Depending upon how often the image is used, it may be saved in the cache. Cache is managed by discarding images that

are oldest, or some other cache management scheme, and if no longer in cache the image can be regenerated from the parameters stored at the URL.

To write software that makes images, a user would need to know or have access to C++, Java Script, or the like, and an understanding graphic file formats, including an understanding of what a URL is and how the Internet handles a URL request. References: (1) R. Fielding, et al. HTTP Working Group Internet-Draft, HYPERTEXT TRANSFER PROTOCOL - HTTP/1.1, <draft-ietf-HTTP-v11-spec-06>. (2) James D. Foley, et al. "Computer Graphics: Principles and Practice", Second Edition in C, Addison-Wesley Publishing Company, copyright 1996, 1990, ISBN 0-201-84840-6. 1175 pages.

Referring to Figure 25 and Figure 26, graphics server 350 functions as follows. In step 380, the server receives the URL of a graphic request, such as from step 375 (Figure 24). In step 381, the URL string is hashed to obtain a hash number that, in step 382, is used to look for an image in cache with a matching hash number. If, in step 383, that matching image is found, in step 384 the image is served in response to the request. However, if that matching image is not found, the image must be regenerated. In step 385, the URL from the request is parsed to obtain the graphic

parameters which, in step 386, are used to construct the graphic server objects, using default parameters for any necessary parameters missing from the URL. Then, for each object constructed in step 386, in step 387 a background object is constructed, in step 388 the font object is constructed, in step 389 the text graphics are made, in step 390 graphic effects from the URL are applied, and in step 391 the background object is combined with the font, text, and effect objects to create the final image. Graphic effects include blurring, shadowing, transparency, etc. If animation is to be applied, in step 392 the number of frames required is determined, and in step 393 the animation effects selected. These include fade in, fade out, fade in/out, undulation, and slide. Slide includes in, out, loop, up and down. Undulation refers to transition between colors. For animation, a group of still pictures are created in a display sequence. Text is made graphic by creating an image of text and font objects with effects applied. Font objects are parameters that constitute selected font effects, including type, point size, color. A background object is the canvas on which document is painted or created.

Online/Offline Updating of Web Site

Uninterrupted online and offline updating of a web site is provided by QuickPlace such that other clients are not interrupted when a client is operating in draft mode. As was previously described with respect to Figures 14 and 15, a QuickPlace user may change a page by creating a draft copy when in edit mode, work with the draft copy, and save it without disturbing others who are using the original page. When finished, the user client publishes the draft page, which then replaces the original copy. Others viewing the original copy continue to use that original copy until they refresh, at which time they are served the newly published page. Normal conflict resolution may be provided.

Referring to Figure 27, a QuickPlace place, or object, includes one or more rooms, shown here as main room 271, lounge 272, and subroom 273, arranged in a hierarchical structure. In accordance with the preferred embodiment of the invention, interactive modification by way of a client browser of a such a site is allowed through the use of server side commands that respond to browser functions. These server side commands are QuickPlace commands 160 that are built on top of Dominos 132 (Figure 6). For example, as will be more fully described hereafter, a create room command exists in QuickPlace.

Referring to Figure 28, the manager of lounge room 272 creates subroom 273. To do so, the user selects the create room command as follows: he first selects 'customize' 276 in the side bar; this brings up another submenu including rooms 277 which, when selected bring up still another new screen 278 which includes parameters for creating room. Once those parameters are selected, the user clicks 'done'. In response, server side processing occurs. First, server 132 creates a subroom, which is another database 283, updates the forward pointer 282 to subroom 283 from lounge room database 281, with the user as manager and as only user. User may then goes into the display of subroom 273 to update the room description characteristics, and optionally add other users authorized to do the same.

Express Mail Label

Inheritance of Aesthetics

Referring to Figure 29, a QuickPlace manager of a room has flexibility in determining the aesthetics of the user interface 260 of that room. The manager, through the QuickPlace hierarchical inheritance feature, can decide that all subrooms of this room look a certain way. This is done by first changing this room, then all subrooms thereafter created from this room will inherit its aesthetics, and

those subrooms previously created, but which have not had their aesthetics previously explicitly changed, will also inherit those changes. Once a subroom is explicitly changed, however, it no longer inherits changed aesthetic characteristics of its parent.

QuickPlace offers a preset gallery of web presentation formats, from which the manager of room can choose and then modify the one he chooses. A room 260 presentation includes a page, or work area, 261, an action bar 262, a side bar 263, a path bar 268, and persistent links 269. In accordance with an aspect of the invention, a user may upload a user configuration specification by which the design and appearance of a room presentation may be modified.

Referring to Figures 31 and 32, the processes for customizing room decoration and for decorating a room being entered are set forth.

For room decoration customization, in step 301, the user selects the action customize decorate. He is presented with two options: in step 302 to select a preset theme, or in step 303 to tweak (that is, modify) an existing theme.

Referring to Figure 31, if the user selects option 1, in step 304 the server provides a list of themes stored in gallery resource database 292. In step 305, the user selects the theme from the list, and in step 306 the server copies the selected theme settings to the setting document for the room being decorated. In step 307, the server sends a new set of instructions to the user browser which represent the decoration, or aesthetic characteristics, of the room being decorated.

If the user selects option 2, in step 308 the server provides a list of options, or user interface (UI) elements, to tweak. In step 309, the user selects a UI element (or option) and in step 310 the server displays the possible tweakable, or modifiable, attributes for that element. In step 311 the user selects the attributes and indicates the changes. In step 312 the server modifies the room settings document, and in step 313 refreshes. This results in step 307, in which the server sends the new set of decoration instructions to the browser.

For decorating a room being entered by a user, in step 321 the user enters a request to enter a room. In step 322, the server accesses the current room settings document for the room to determine if settings exist. If so, in steps

323 and 327, those settings are returned to the user browser for displaying the room. If not, in steps 323 and 325 ancestor rooms are traversed, looking for a room with settings and, if not found, in step 326 the server accesses the gallery resource database to obtain default decoration settings for return to the user browser.

Collaboration

In accordance with another aspect of the invention, a user interface and data storage model is provided whereby a published piece of content on the web can be simultaneously edited by an author and saved periodically by the author, yet the published version doesn't get touched (modified) by viewers until the author is ready. This achieved by having the server manage both the published version and a draft under construction such that the published version is made available for viewing by some users while another user is modifying the draft, and then merging the two versions into a new published version. The server delivers to the user making the modifications a complete set of offline code, including a replica of the server and of the draft while maintaining a healthy replica with respect to security.

Replication

Referring to Figure 33, in accordance with the preferred embodiment of the architecture of the invention, there are provided a server side segment 163 and a client side segment 159. Server side elements 163 are used by a website developer to prepare a Domino application to work offline. Client side elements 159 are those elements which must be installed to an end-user's machine in order for the Domino application to run. Domino runtime services (DRS) 163 comprises utilities that help assemble an appropriately minimized set of files 409 to include for producing a client-side 159 install which can make a Domino web application run disconnectedly on a client machine. A DRS client 159 comprises a stripped won locally running Domino server, including service manager 422, dedicated to running only one Domino web application. This local server needs far fewer feature elements than a full online server. Client side elements are installed on the end user's computer by downloading and running the install program 422 that is built using DRS server side 163 elements.

A software system environment is organized into segments 159, 163, which are major configuration groupings.

Within segments are items, which are software items which are environmental or deliverable items. QP server 163 is a deliverable item. Netscape 4.5 browser 101 is an example of an environment item. The sum of the segments define a system. Deliverable items include software units.

First, with respect to the server side segment 163. Environment items include, for example, WIN NT 401 as the operating system, mail router 404, and QuickPlace server 163, Domino server 132, HTTP task 134 for serving web pages, and Domino extension 160 for opening a Notes database. QP server extensions 160 are inside the Domino box, and an item within that box is the ID assignment utility 402.

Four, primary architectural components comprise the Domino Offline Services, and support the offline environment ID assignment extension. These are first (I), ID assignment utility 402, second (II) Domino runtime services (DRS.nsf) 403, third (III) DRS download control 407, and fourth (IV) footprint 409. The first architectural component is the ID assignment utility 402, which is a sub-task within the QP utility 160 involved in supporting the off line environment. The second architectural component is the ID certifier repository 403, which is a small DLL for executing the appropriate setup. The third architectural component is

download control 407. The fourth architectural component on server 163 unique to going offline is a collection of compressed Notes DLL's, compressed for downloading, referred to as the "footprint" 409. These four architectural components, together comprise the Domino extensions required for taking offline a server, such as a QuickPlace server, including code, data, and a fully functional security model, to enable that server to function offline in either a web browser or Lotus client environment.

QuickPlace server (QPS) 163 keeps a copy of a Notes certifier identifier in DRS.nsf, and that certifier ID supports the dynamic creation and generation of Notes user ID's. Main.nsf is a QP database in which is maintained special pages, or scenes, dedicated to offline operation. These offline scenes trigger the DRS download controls, including active x and Netscape plug in. Active x is the name of a component model that extends Internet browsers. Java applets, active x comments, and Netscape plug ins are different ways to augment browser function with code that the browser executes.

When a user visits an offline page, the download function is triggered. Go offline scenes 405 are QP 163 pages in server 163 that provide instructions on how to go

offline, and provides the download control with right parameters for downloading. In this context, down load includes downloading web site code, which is the minimum software needed for client segment 159 to run offline (that is, detached) from server 163.

In operation, a user at a browser visits a web site, encounters a download page, or offline scene, and activates download control with parameters specific to the web site being visited. In response, download control 407 loads to the client environment a set 435 of .nsf files for incrementally executing download functions. Four steps 435 are executed to incrementally install at client 159, from footprint 409, the agent files, full text index files, and so forth required to operate browser 101 offline.

The first step, step 411, determines which Notes ID is to be utilized. If the client is a browser (not a Lotus Notes client), this step 411 triggers ID assignment utility 402. ID assignment utility 402 obtains the certifier ID from repository 403, and from that generates the Notes ID which it passes to download control 407. With that Notes ID, download control selects and downloads the appropriate footprint files, or DLLs, 409.

The second step, step 412, an instance of download control 407 on the client side 159 uncompacts the downloaded footprint files to generate installer files 413.

The third step, step 414, executes the installer files which prompts the user for information, installs several execute files, including nhttp.exe; installs several DLLs, and prepares the Notes.ini files. At this point, a fully installed file system exists on the client, including the DLLs that comprise the runtime of Notes. nhttp.exe 414 is the same web server component code as http 134 on the server side. This file 414 is the program that serves web pages, together with the Notes run time DLLs that support it, and these accumulate in directories at the client in preparation for step 416, when Main.nsf 193 will be downloaded. This file 414 is a critical component, and is, in effect, the mini server task, including the same code that comprises the web server component at the server. As a result of this identity, both online and offline pages exhibit the same behavior to the user. Thus, browser 101, when operating on line, accesses web server 134, and when operating off line, accesses web server 414. In both cases, the behavior is the same.

Components installed by step 414 (step 416 brings down

the data) include a base grouping 420, including the service manager 422 execute (.exe) file, the http component 423,previously described, update component 424, and replicator component 421. These are the key, or base grouping, of components brought down in step 414 and activated every time there is an offline configuration. Because replicator 421 is required to be kept up to date, and http 423 required to be viewable in browser, update service 424 keeps the other two (421 and 423) aware of each other's actions. When replicator 421 brings new documents to a database, it doesn't refresh the views. Because http 423 assumes the views are refreshed, update 424 sees that they are.

If more are services needed, step 414 may activate additional services 429. Other services 429 possibly includes mail dispatch, full text index, agents, Lotus script, Java back end classes, and so forth. These are controlled by the service manager 422. Service manager 422 is the program that the user interacts with after download is completed.

In the fourth step, step 416, first time replication brings down main.nsf by turning on the Notes DLLs with the Notes ID that was previously downloaded, and then running

them as though they were a Notes client for replication. An advantage of this approach is that the replication and security model of this web client is exactly the same as that for a Notes client.

At this point, a user can start using this offline version of the web site.

Service manager 422 is a utility a user can launch from a desk top, from start menus, and from a driver tray, and behaves like a system driver. Such a driver, if clicked on in tray, gives in a pop up menu such options as online, offline, synchronize now, edit synchronization properties. Synchronization properties is a tabbed dialog (or, in QP, a web page), which is a piece of user interface for setting the replication schedule and any required user filters. Examples of user filters include filters on how much data to bring down: such as, days worth, no more than x bytes, or stop after an hour. Service manager 422 also launches other tasks as needed by service manager 422 without user intervention. Thus, service manager 422 is an easy interface to a customer, and is a smart manager of mini server tasks.

Architecturally, main.nsf 196 includes a cross link

component - a piece of html or some kind of web element that is highly customized to a specific application and which is consistent in role to main.nsf 193, giving the user a visual indication of version of main.nsf is being used (online 193 or offline 196). With one click at the browser 101, the user can move to the other main.nsf version. This is useful after main.nsf is downloaded first time. Web browser 101, after download refreshes main.nsf 196, shows a go offline link that sits in top bar of the application, such as a QuickPlace application, view panel and when clicked switches the browser to the client http task 414 from the server http task 134. Thus, the off line url is for the client machine 159, and the on line url is for the server machine 163, and these url's only differ in the server name part of the url.

At this point, the user uses a personal copy of a web site. Replication is triggered thereafter by schedule or at user request.

Download is done using http protocol, transported on TCP/IP port 80. Notes replication is transported on TCP/IP port 1352. This arrangement allows for a fire wall between client 159 and server 163. This fire wall must be configured by the administrator to allow transfer because http is not a secure protocol. A security variation uses

https for download, and encryption is turned on for replication protocols.

Uninstall 417, when executed, cleans everything brought down except for a version of download control 407 cached at client 159.

In accordance with the security model implemented by the preferred embodiment of the invention, the user ID is used by all processes, particularly the replicator, to assure that a user is not able to access anything to which the user is not entitled. Main.nsf offline 196 only includes what the user is allowed to see when accessing main.nsf 193 at the server.

Forms Creation and Maintenance

Referring to Figure 34, it is a further aspect of the invention that there is provided to a user at a browser the ability to create and maintain forms 331, 333, and incorporate them into a room. Any page, or document, in the system is created using a form, which is a layout that identifies page content in the terms of fields 335, 336, which map to items 337, 338 and 339, as illustrated. One

form may be used to create more than one document, as is illustrated by field 335 of form 331, which is used in defining item 338 in document 332 and item 339 in document 334. A form provides fields and layout.

Referring to Figure 35, from a browser, a user with manager access to a room can create a new form by defining the fields that make up the form: name, field, type. Types are edit box, radio button, check box, attachment, and are the same as field types in Notes or HTML. User (manager) goes to customize page, selects forms page, and selects the new form button. A screen comes up asking for the name of form, the fields in the form, and the workflow attributes of that form. The user adds fields by selecting add fields, in response to which a list of field type appears. The user then chooses field type (edit box, check box, rich text, attachment, author, date, etc.). Behind the scenes, a note 331 is associated with the form and a note 342 is associated with each field. A field note 342 is associated with each field 341 in the form note 331. A QuickPlace form object 343 is a form note 331 and associated field notes 342. Depending on the type of field selected, each field note 342 has different attributes.

After a user has created a QuickPlace form, to put this

form into action, this or some other user creates a page using this form. For example, an inventor may come into the place where this form is stored and address the form by the name given by the author. The QuickPlace system creates a page using the definition of the form which includes the field notes 342, the layout from the form note 341, and workflow attributes. A document is a type of note. Other types of notes are view, folder, agent and form. When user hits "done", browser sends a page with data to server which creates the resulting document 332. The page 332 presented to the user by the server can be used by the user, who fills out the values for the fields 346. Thus, a document is created by a server from a browser entered page.

By providing an interactive facility in QuickPlace for accepting and storing form data in an object store, which is a database or page, at the server, there is provided a way for a client at a browser to create forms which include a definition of where and how data should be stored, and which enables others to manipulate the form.

A Notes form is a Notes object that is used as a form by a notes designer and user. In QuickPlace, the client is the browser, so Notes forms need not be used. Rather attributes of a QuickPlace form are stored in a note using a

record in a database.

Notes offers a number of objects, including the Lotus Notes Product, which includes databases, which includes notes, which are view, folder, form, agent, and document notes. Applications are built on this model.

In Lotus Notes, a view is a collection of documents. A folder is a collection of documents. A form is an object that contains fields and layout, and is used for the creation of documents. A document is a record in a data base. An agent is a set of instructions or logic that perform a given function, such as a macro defined by the agent creator.

In QuickPlace, Notes objects are fine tuned to support a Notes client, where the client is a web browser. In accordance with the QuickPlace model, there are provided places, which contain rooms, members, forms and fields, and where a room contains folders, and folders contain pages.

To build or implement this QuickPlace model on top of Notes object model a Place becomes a file system directory (which is not a notes object), a room becomes a note (of type database), a folder becomes a note (that stores folder

attributes) + folder or a note (a store of folder attributes) + view, depending upon which is most convenient. A page is a note (of type page, containing data associated with page), a member is a note (of type member) in the member's database, a form is a note (a form note), and a field is a note (a field note). Note types are page, member, folder, field, form, and room.

Workflow

In accordance with another aspect of the invention, workflow functionality is provided at a browser. A user fills out and submits a form on the web. It then travels to individuals in a prescribed set and order. The form is submitted via HTTP to a server. The server creates other pages and elements to be delivered back to browser. These elements store state information about the pages submitted to track who has had, and who needs to get the form and in which order. When a browser defines a workflow document, E-mail notifications are sent by the server to each person along the path. In this way, a user is enabled to define a workflow document with a browser that can be executed by many other users with browsers.

In this workflow process, after an author has created a form with several fields, workflow logic is added. When creating the form, author decides type of work (editor in chief, approval cycle, multiple editors). Editor in chief allows specification of a person to approve the page before it will be published. Approval cycle specifies a list of individuals that the page is to go to serially; that the pages that are still pending be placed in a holding folder; who can edit the page after final approval - final reviewer or page creator; and who can read the page after approval. In creating and using workflow forms, the QuickPlace is accessed by room, then room options, forms, create new form, workflow buttons are selected to bring up a screen with all of the options.

Workflow attributes of a QuickPlace form created by a browser are stored with the form note, enabling the QuickPlace server to properly control the workflow routing. A page created from the form includes the security items h_readers and h_authors, each of which includes a list of one or more people. Readers can read, authors can edit, thus allowing security to be at the page level. h_cycle list provides a list of people in the approval cycle. At end of approval cycle, the page can be directed to a specific folder. Readers and authors are security items

stored with every page; other attributes (than these security attributes) are strictly workflow attributes, such as current approver, which is used to determine where the page goes following current approval, and so forth.

In accordance with a preferred embodiment of the invention, five different workflow processes are supported. These are standard workflow, simple submit workflow, approval cycle workflow, editor-in-chief workflow, and multiple editors workflow.

First, standard workflow. This workflow process allows the user to either publish a page immediately or publish with options. This is the workflow process that is used for the standard forms which ship with QuickPlace and is the default workflow process for custom forms. Options available to a user who creates a page which uses this workflow are (1) publish immediately, or (2) publish with options.

Publish Immediately: If the current context is a folder, the page is published and put in that folder. Otherwise the user is asked where it is to be put (in a folder or in the table of contents (TOC)). On a custom form, the creator of the form can also indicate what folder

this type of page should be put in, in which case the user is not asked for this information.

Publish With Options: The user can publish the page or save it as a draft. If it is a new page, the user is allowed to select where it is to be put (folder or TOC). If a previously published page is being edited, the user can choose to revert to the published version. If it is being published, the user can select from none to all of the following options:

Notify: allows the user to send an E-mail with a link to the current page and some explanatory text.

Add Authors: allows the user to specify who is allowed to edit this page (if not selected, only the author can edit the page).

Restrict Readers: allows the user to specify who is allowed to read this page (if not selected, everyone can read it).

Add to Calendar: allows the user to specify a date and time so the page will display in the calendar as well as the folder in which it is put.

Move: this is only available during edit (not for new pages) and allows the user to move the page to a different folder or to the TOC.

Second, simple submit workflow. This workflow process allows the creator of a custom form to indicate what to name the submit action (for example, it may be named "Done"). This action then works exactly like Publish Immediately, described above.

Third, approval cycle workflow. This workflow process allows the creator of a custom form to indicate up to four approvers for a sequential review workflow, and to choose among the following five form creation options: (1) In Process Folder (optional), which specifies what folder the page will reside in during the approval cycle (see below for what happens at final approval). (2) Final Editor, which specifies whether the author or the final approver is the person allowed to edit the page after approval is complete. (3) Final Readers, which specifies whether everyone, or just the author and approvers, are allowed to read the page after approval is complete. (4) Final Location, which indicates whether the final approver determines where the page goes (folder or TOC), or that the page will simply be moved to a specific folder upon final approval. (5) Rejection

Destination, which specifies who the next person in the workflow is if any reviewer rejects the page, either the author or the previous reviewer.

In approval cycle workflow processing, two form use options are available. First, when a user creates a page with this form, a Submit action is provided. Selection of Submit brings a scene where the user can select Submit or Submit with Explanation, and if In Process Folder (see above) is not indicated, the user chooses where to put the page (folder or TOC). Selection of Submit automatically sends an email to the first approver, with a link to the page, notifying there is a page which needs review and making the first approver the sole editor of the page. Submit with Explanation allows the author to add some explanatory text to the email. Second, when a reviewer has reviewed and possibly modified the page, a Submit action is also available. Selection of Submit brings a scene similar to the one described above, except that options to Reject, or Reject with Explanation are also presented. Rejected pages are moved to the Rejection Destination (described above). E-mail is sent to the next reviewer, and the next reviewer becomes the sole editor of the page. If the current reviewer is the final reviewer and Final Location (above) is set to final approver, the destination for the

page (folder or TOC) is selected.

Fourth: editor-in-chief workflow. This workflow process is essentially a single approver workflow. The options described above are automatically set as follows and, with one exception, are not modifiable by the custom form creator. If the form creator wants different options, an approval cycle workflow process may be used with one approver. The option defaults for editor-in-chief workflow are as follows. (1) In Process Folder (optional) is the only modifiable option and defaults to none. (2) Final Editor: the page author is the editor (if the page is re-edited, it goes through the approval again). (3) Final Readers: everyone can read. (4) Final Location: approver decides. (5) Rejection Destination: the page author.

Fifth: multiple editors workflow. This workflow process is exactly like the Standard Workflow, except that instead of the author indicating who can edit the page, all people with author access to the room are allowed to edit the page.

Forms Upload

In accordance with a further object of the invention, a user can decide the appearance of an HTML form using an HTML editor and save to file. To make it a QuickPlace form, the HTML file is uploaded through a bucket, and the server converts it into a QuickPlace form. In this manner, externally designed HTML forms can be incorporated into a QuickPlace site. Once a form is a QuickPlace object, it has additional features. This form may then be offered to authors to create new documents. It is available in the room in which the user is operating, and can be adapted to workflow.

Upload Agents

In accordance with a further object of the invention, a user with a browser can write and upload to a server logic that will react to the submission of a page. By way of example, a form is developed to allow people to submit patent applications. This is put in QuickPlace patents place. To manage the creation of these pages, logic is added to the form via Lotus script or Java. For example, whenever a page is submitted, and the description includes "workflow", then the page is filed in a folder for "workflow". In this way a user may independently develop

logic and then upload it to his QuickPlace place.

TABLE 1: QUICKPLACE COMMANDS

	(See Table 2 for definitions of the fields listed as arguments.)
h_ChangeACL	Creates, removes, or modifies new access to a room for a list of users. Arguments include h_SetEntryNames, h_SetEntryTypes, and h_SetAccessLevels.
h_CheckHuName	Checks if a given QuickPlace Name h_Name is available. Two QuickPlaces cannot have the same name.
h_CreateFolder	Creates a QuickPlace folder. Arguments include the user visible name of the new folder and its style, whether standard list, headline, slide show, response list, or ordered list. Arguments include h_Name, h_FolderStyle, h_FolderStorage.
h_CreateGroup	Creates a group composed of the specified users. Once created, the system mains four fields for each group, including: h_Name, h_SystemName, h_Members, and h_Type.
h_CreateOffice	Creates a new QuickPlace (formerly known as an office). This means creating a directory for the QuickPlace, creating the tope level room specified (and any subrooms specified by that room), and creating a manger for the QuickPlace. Arguments include h_Name, h_AreaType, h_AeName, h_UserName, h_Password, and h_EmailAddress.
h_CreateRoom	Creates a QuickPlace room as a subroom of the current room with the specified aesthetic set and of the specified room type. Arguments include h_Name and h_AreaType.
h_CreateUsers	Creates members in a Place. Arguments include: h_SetUserNames, h_SetPasswords, h_SetEmailAddresses, h_SetNewLevels, h_SetEntryNames, and h_SetAccessLevels.

h_DeleteFolder	Deletes the current folder. No arguments.
h_DeleteGroups	Deletes one or more groups from a QuickPlace. Argument: h_SetGroupNames.
h_DeleteOffice	Deletes one or more QuickPlaces. Argument: h_HaikusToDelete, which is a textlist of QuickPlaces to be deleted.
h_DeletePages	Deletes pages indicated in the argument h_SetDeleteList.
h_DeleteRoom	Deletes the current room. No arguments.
h_DeleteUsers	Deletes the specified users from the system. Argument: h_SetUserNames.
h_MoveRoom	Moves a room within the room hierarchy. Argument: h_NewRoomName, which is the nsf name of the room's new parent.
h_ReorderRoomArea	Used to change a room name, set the mail delivery into a specific room, show or hide the calendar, security information to reader and authors, and to reorder the table of contents.
h_SendMail	Sends an email to the indicated recipients. Arguments include h_SetPublishEmailAddresses, h_SetIP PublihsEmailSubect, and h_SetPublishEmailMessage.
h_SetAesthetic	Sets aesthetics on a room by copying from a stored aesthetic set, copying from the parent's aesthetics, or resetting a room to use is parent's aesthetics. Arguments include h_AestheticCmd and h_AeName, which is the name of the aesthetic set stored in the QuickPlace Resources (Resources.nsf).
h_SpellCheck	Spell checks the indicated test. Argument: h_PageText.
h_UpdateGroup	Changes the name or membership list of a group. Arguments include h_SystemName, h_Name, and h_Members.
h_UpdateUser	Changes a user's information. Arguments include h_SystemName, h_UserName, h_FirstName, h_LastName, h_EmailAddress, h_SetPassword and h_Phone.

TABLE 2: FIELDS OF QUICKPLACE OBJECTS

BASIC

h_Type:	The type of object. One of: page, folder, room, subroom, error page, roomType, h_Member, h_Group, h_Form, and h_Field.
h_Name:	The user visible name of this object.
h_SystemName:	The name of this object as known to the system.
h_isSystem:	1= This is a system object.

ROOM

h_HaikuName:	The name of this Place.
h_AreaType:	The name of the template used to create this room.
h_AreaParent:	The name of the parent database.
h_ShowSecurity:	if h_SetSecurity=1, servers sets h_ShowSecurity to 1.
h_SetCalendar:	1=Show the Calendar in the sidebar.
h_SetTutorial:	1=Show the Tutorial in the sidebar.
h_SetSecurity:	1=Show Security in the sidebar for Readers and Authors (if they select Security in this case they will see only their own information).
h_MailDb:	The name of the database that receives email addressed to this Place.
h_LastAttachmentDirectory:	Last directory used when getting attachments.
h_DirtyAesthetics:	Number which indicates which items should be checked (once a part of the aesthetics has been tweaked, the check mark indicates that part has already been changed).
h_AreaHasAesthetics:	1=This room has its own aesthetic settings.

FOLDER

h_FolderStyle:	One of standard list, headline, slide show, response list, and ordered list.
h_FolderStorage:	The name of the folder as known to the system.
h_CanAddPages:	1=All authors can add pages to this

folder.
0=Only managers can add pages to this folder.

FIELD

h_ContainerUNID: The UNID of the Form which contains this field.

h_FieldType:

h_Subject	Subject
h_TextInput	Simple text (<input>)
h_RichText	Rich text
h_Attachments	Attachments
h_TextPopup	Text select list (<select><option>)
h_NamePopup	Select listing members of the of the QuickPlace
h_DateControl	Date field with date picker widget
h_TimeControl	Select lists for hours, minutes, AM/PM
h_CalendarControl	Includes date and time controls and a duration field
h_Serial	A unique number
h_Static	Static text and/or graphic
h_DocAuthor	Original author of the page
h_DocCreated	Creation date of the page
h_DocModified	Modified date of the page
h_DocSize	Size of the page

h_FieldLabel Instructional information, only displayed in edit mode, that might be useful for someone editing this field.

h_FieldFormat: Indicates formatting options for

h_CaledarControl	Whether date, time and/or duration are included
h_Subject	Whether banner is displayed or optional.

h_FieldIsRequired 1=The field is required and the user will be prompted if it is not filled out.

PAGE

h_Form: The QuickPlace form used to create this page.

h_PageType: Usually null. Could also be h_Response, h_Revision, or h_Mail.

h_NameIsBanner: 1=The name of this page should be displayed as a banner.

h_Originator: The creator of this page.

FORM

h_FormDescription: The description of the form that appears in the "New" scene.

h_WorkflowType:

h_ApprovalCycle	Allows 1-4 approvers and some other options
h_EditorInChief	Allows 1 approver and fewer options.
h_MultipleEditors	All members of QP can edit pages created with this
h_Standard	None of the above

h_SetPageComponentsView: Should=h_FieldDefinitions.

h_IsUserDefined: h_True means this is a custom form.

h_PublishInFolder: UNID of the folder + "|"+h_FolderStorage name of the folder.

MEMBER

h_Password: This member's password. Encrypted with @Password.

h_FirstName: This member's first name.

h_LastName: This member's last name.

h_PhoneNumber: This member's phone number.

h_EmailAddress: This member's email address.

GROUP

h_Members: The list of members who belong to this group.

PUBLISHING

h_IsPublished: 1=This object is currently published.

h_IsHidden: 1=This object is not shown to the user.

h_SetReadScene: The name of the default scene (subform) to use when viewing this object.

h_SetEditScene: The name of the default scene (subform) to sue wen editing this object.

h_PublishedVersionUNID: If this object is being edited and

		the current object is the draft version, the UNID of the published version of this object.
h_DraftVersionUNID:		If this object is being edited and the current object is the published version, the UNID of the draft version of this object.
h_LastTimePutAway:		The last time that this object was changed: Published or Saved under construction.

LOCATION

h_FolderUNID:		The name or UNID of the Notes Folder where this page resides.
h_IsInToc:		1=This object is shown in the Table of Contents (sidebar).
h_CurrentPosition:		The position of this object with respect to other objects in the collection.
h_SetParentUNID:		If this is a child or response object, the UNID of the parent object.

SECURITY

h_Readers:		If this object is protected from readership, the list of names, groups, and or roles that can read this object.
h_Authors:		If this object is protected from authorship, the list of names, groups, and or roles that can author this object.

WORKFLOW

h_WorkflowState:	h_New	Created but not yet submitted for approval.
	h_Submitted	Has been submitted and it is being reviewed.
	h_Published	Has been approved.
	h_Rejected	Has been rejected.
h_SetNextStageUser:		The name of the next person in the workflow cycle.
h_CurrentApprover:		A number designating the current person in the workflow cycle. 0 means the Originator. The list of persons associated with the workflow cycle is stored in the form used to create this

		page. See h_Form.
CALENDAR		
	h_CalendarDate:	The calendar fields are present when the page has been added to a calendar. The date that his object should appear in the calendar.
	h_CalendarTime:	The time that this object should appear in the calendar.
	h_CalendarDuration:	the length of time associated with this object when viewed in the calendar.

ADVANTAGES OVER THE PRIOR ART

It is an advantage of the invention that there is provided a system and method which enables a user to accelerate productivity by easily creating and maintaining collaborative spaces on a network.

It is an advantage of the invention that there is provided a system and method which enables a user to customize security attributes of rooms in a collaborative space including a hierarchy of rooms.

It is an advantage of the invention that there is provided a system and method for uploading documents which

involve the more intuitive process of dragging and dropping the document to a spot on the site where it is to be rendered.

It is an advantage of the invention that there is provided a system and method for managing security of files in a hierarchical database such that security may increase or decrease at a child with respect to its parent.

It is an advantage of the invention that there is provided a system and method for enabling browser creation and modification of web sites.

It is an advantage of the invention that there is provided a system and method for enabling uninterrupted online and offline updating of a web site.

It is an advantage of the invention that there is provided a system and method whereby the aesthetic characteristics of a parent room may be inherited by its children rooms.

It is an advantage of the invention that there is provided a system and method for enabling a browser to define a work flow document and process.

ALTERNATIVE EMBODIMENTS

It will be appreciated that, although specific embodiments of the invention have been described herein for purposes of illustration, various modifications may be made without departing from the spirit and scope of the invention. In particular, it is within the scope of the invention to provide a computer program product or program element, or a program storage or memory device such as a solid or fluid transmission medium, magnetic or optical wire, tape or disc, or the like, for storing signals readable by a machine, for controlling the operation of a computer according to the method of the invention and/or to structure its components in accordance with the system of the invention.

Further, each step of the method may be executed on any general computer, such as an IBM System 390, AS/400, PC or the like and pursuant to one or more, or a part of one or more, program elements, modules or objects generated from any programming language, such as C++, Java, Pl/1, Fortran or the like. And still further, each said step, or a file or object or the like implementing each said step, may be executed by special purpose hardware or a circuit module

designed for that purpose.

Accordingly, the scope of protection of this invention is limited only by the following claims and their equivalents.

CLAIMS

We claim:

1. A system for serving graphics for display at a browser, comprising:

a graphics server;

a database;

a cache;

said server being operable for storing graphics and text received from a first browser as an image in said cache and said parameters for said image in said database; and

said server being further operable responsive to a request from a requesting browser for selectively serving to said requesting browser for display said image from said cache and, if not found in said cache, for generating anew said image from said parameters stored in said database.

Copyright © 2000 IBM

2. The system of claim 1, said graphics server being operable for storing in said database, and making available to any of a plurality of browsers, parameters for reconstituting a plurality of screen display images.

3. A screen display for publishing a document within collaboration space, comprising:

a work area;

a side bar;

said side bar including a layout button;

said work area, responsive to selection of said layout button, being operable for receiving a title for display in said side bar as a graphic, and for receiving text and graphics for display in said work area.

4. A system for publishing a document within collaboration

space, comprising:

a graphics server;

a screen display including a work area, a side bar including a layout button, and an action bar including insert graphic, publish, cancel, and done buttons;

said graphics server being operable responsive to user selection of said layout button for receiving in said work area a title for display in said side bar and for entering edit mode;

said graphics server being operable in edit mode for receiving in said work area textual content and aesthetic parameters for said document;

said graphics server being operable in response to selection of said insert graphic button for presenting in said work area a create graphic display for user selection of aesthetic parameters for an image;

said graphics server being operable in response to user selection of said done button while in said create graphic display for creating and saving said aesthetic

parameters and displaying said image in said work area in edit mode; and

said graphics server being operable in response to user selection of said publish button for serving said document to said work area in published mode.

5. A method for serving an image to a browser screen, comprising the steps of:

maintaining a cache of cached images;

maintaining a data base of default parameters;

receiving a graphic request locator;

hashing said graphic request locator to form a string number;

examining cache for a cached image with a matching string number, and if found, serving said cached image to said browser screen; and, if not found, parsing said graphic request locator for graphic parameters and constructing from said graphic parameters and

selectively from said default parameters a reconstructed image for serving to said browser.

6. The method of claim 5, further comprising the steps of maintaining a database of graphic parameters, and responsive to receiving a graphic request locator which does not hash to a string number found in said cache, parsing said graphic request locator for the location in said database of graphic parameters for said graphic parameters for reconstructing said image for serving to said browser.

7. The method of claim 5, said step for constructing a reconstructed image further comprising the steps of:

constructing a background object;

constructing a font object;

constructing text graphics;

applying graphic effects from said database; and

combining said background object, said font object,

said text graphics, and said graphic effects to form said image.

8. The method of claim 7, said graphic effects selectively including blurring, shadowing, animation and transparency effects.

9. The method of claim 8, said animation effects selectively including fade in, fade out, fade in/out, undulation, and slide.

10. The method of claim 9, said slide effects selectively including in, out, loop, up, and down.

11. A program storage device readable by a machine, tangibly embodying a program of instructions executable by a machine to perform method steps for serving an image to a browser screen, said method steps comprising:

maintaining a cache of cached images;

maintaining a data base of default parameters;

receiving a graphic request locator;

hashing said graphic request locator to form a string number;

examining cache for a cached image with a matching string number, and if found, serving said cached image to said browser screen; and, if not found, parsing said graphic request locator for graphic parameters and constructing from said graphic parameters and selectively from said default parameters a reconstructed image for serving to said browser.

12. An article of manufacture comprising:

a computer useable medium having computer readable program code means embodied therein for serving an image to a browser screen, the computer readable program means in said article of manufacture comprising:

computer readable program code means for causing a

computer to effect maintaining a cache of cached images;

computer readable program code means for causing a computer to effect maintaining a data base of default parameters;

computer readable program code means for causing a computer to effect receiving a graphic request locator;

computer readable program code means for causing a computer to effect hashing said graphic request locator to form a string number;

computer readable program code means for causing a computer to effect examining cache for a cached image with a matching string number, and if found, serving said cached image to said browser screen; and, if not found, parsing said graphic request locator for graphic parameters and constructing from said graphic parameters and selectively from said default parameters a reconstructed image for serving to said browser.

13. A computer program product or computer program element

for serving an image to a browser screen according to the steps of:

maintaining a cache of cached images;

maintaining a data base of default parameters;

receiving a graphic request locator;

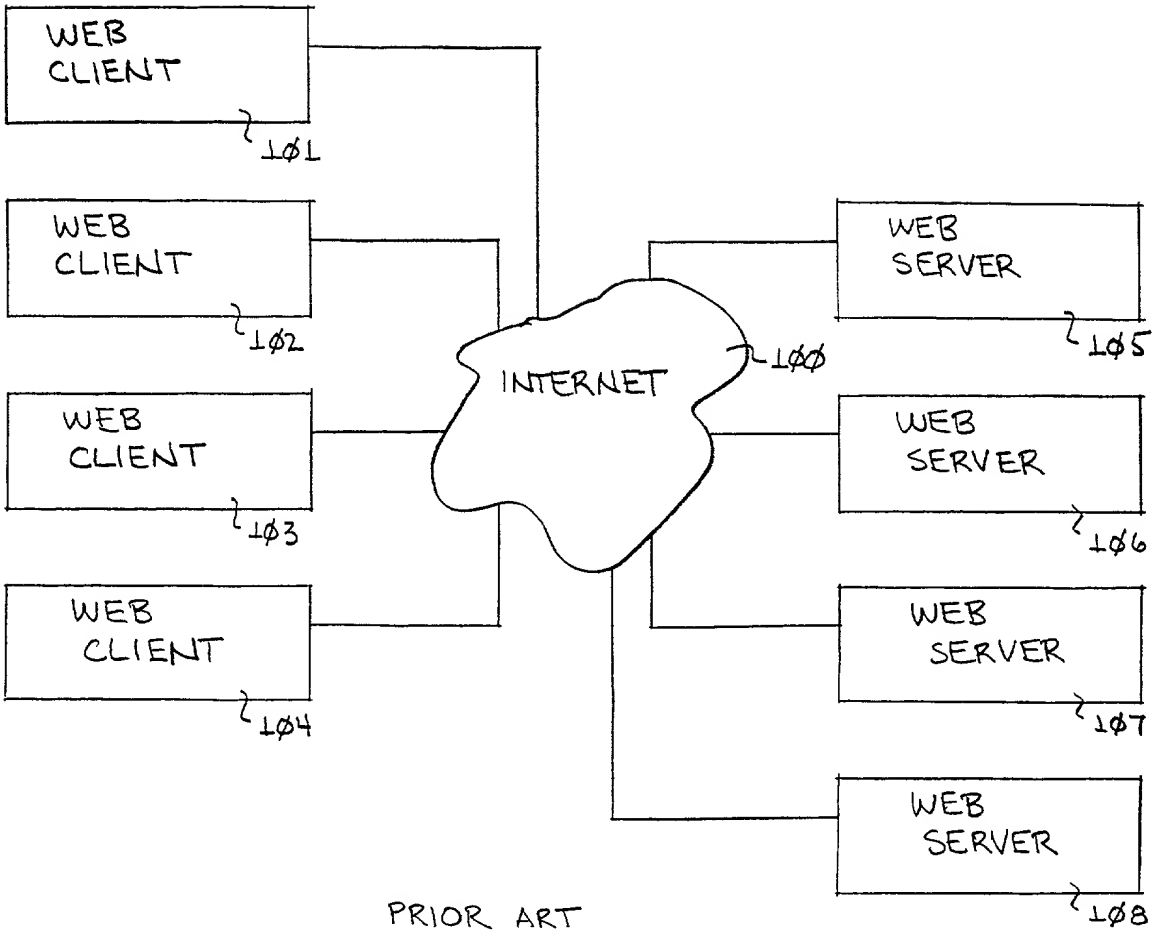
hashing said graphic request locator to form a string number;

examining cache for a cached image with a matching string number, and if found, serving said cached image to said browser screen; and, if not found, parsing said graphic request locator for graphic parameters and constructing from said graphic parameters and selectively from said default parameters a reconstructed image for serving to said browser.

SYSTEM AND METHOD FOR DYNAMICALLY GENERATING VIEWABLE GRAPHICS

Abstract of the Disclosure

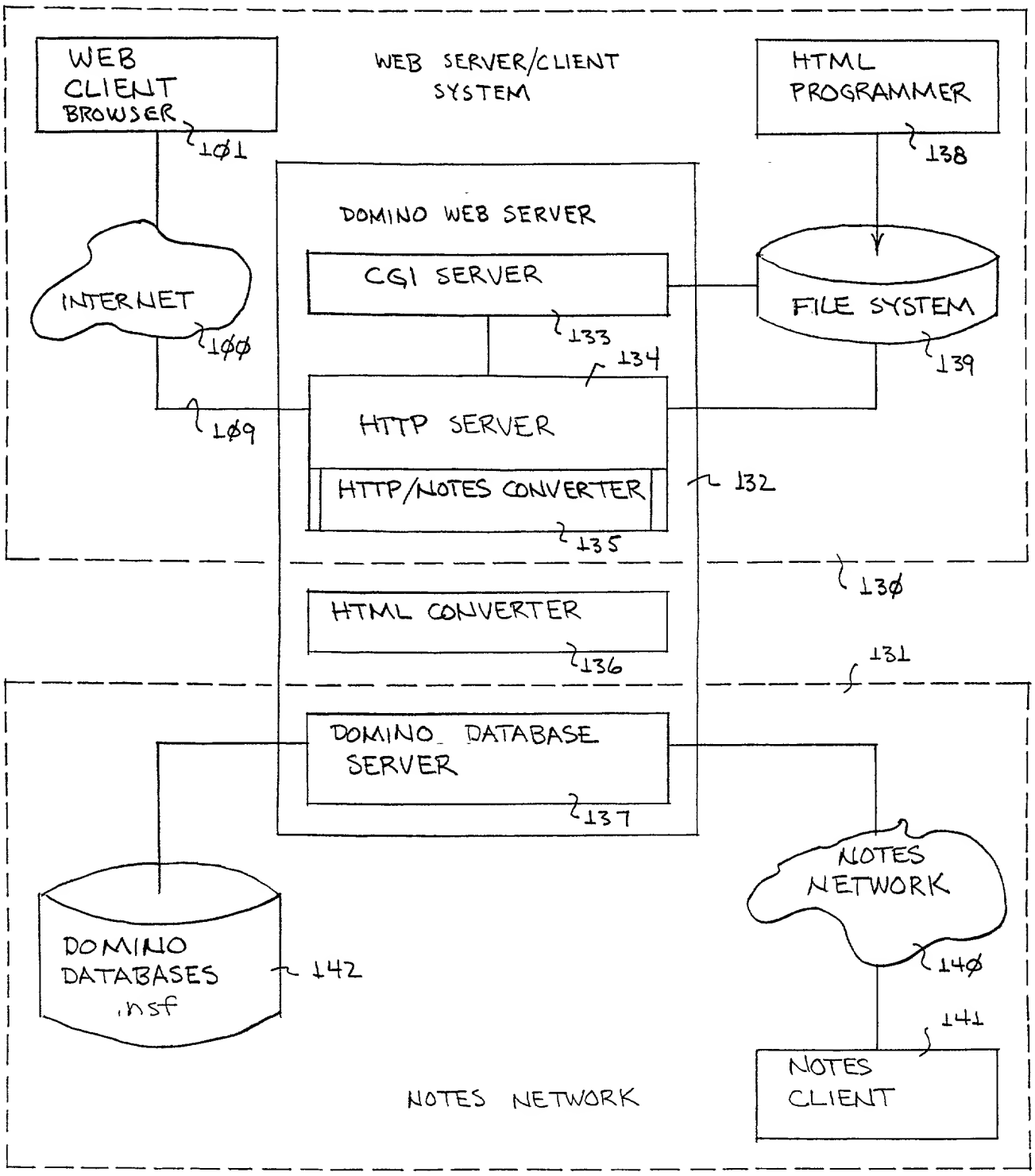
Collaboration space object model provides for a Place consisting of rooms. A room is made up of pages. Folders are used to organize pages. Members belong to rooms, and are those users authorized to access them. Place type controls the creation of a place, including how many rooms it has, for example. Room type controls the appearance and content of rooms. A form manages the display of data notes. A form can contain fields for containing data and employ scripts to process and compute data. A page is the basic vehicle for content. Content is created using an or importing content from an external source. A member is also a data note, and each place contains its own member directory. A place is created and managed from a client browser in on-line mode and in offline mode with respect to a replicated copy of the space. Room security is independently managed, and security and aesthetics characteristics of subrooms selectively inherited. Room navigation and workflow processing is provided, as are forms creation and uploading from browser to server.



PRIOR ART

FIG. 1

Docket No. 50277-2000



PRIOR ART

FIG.2

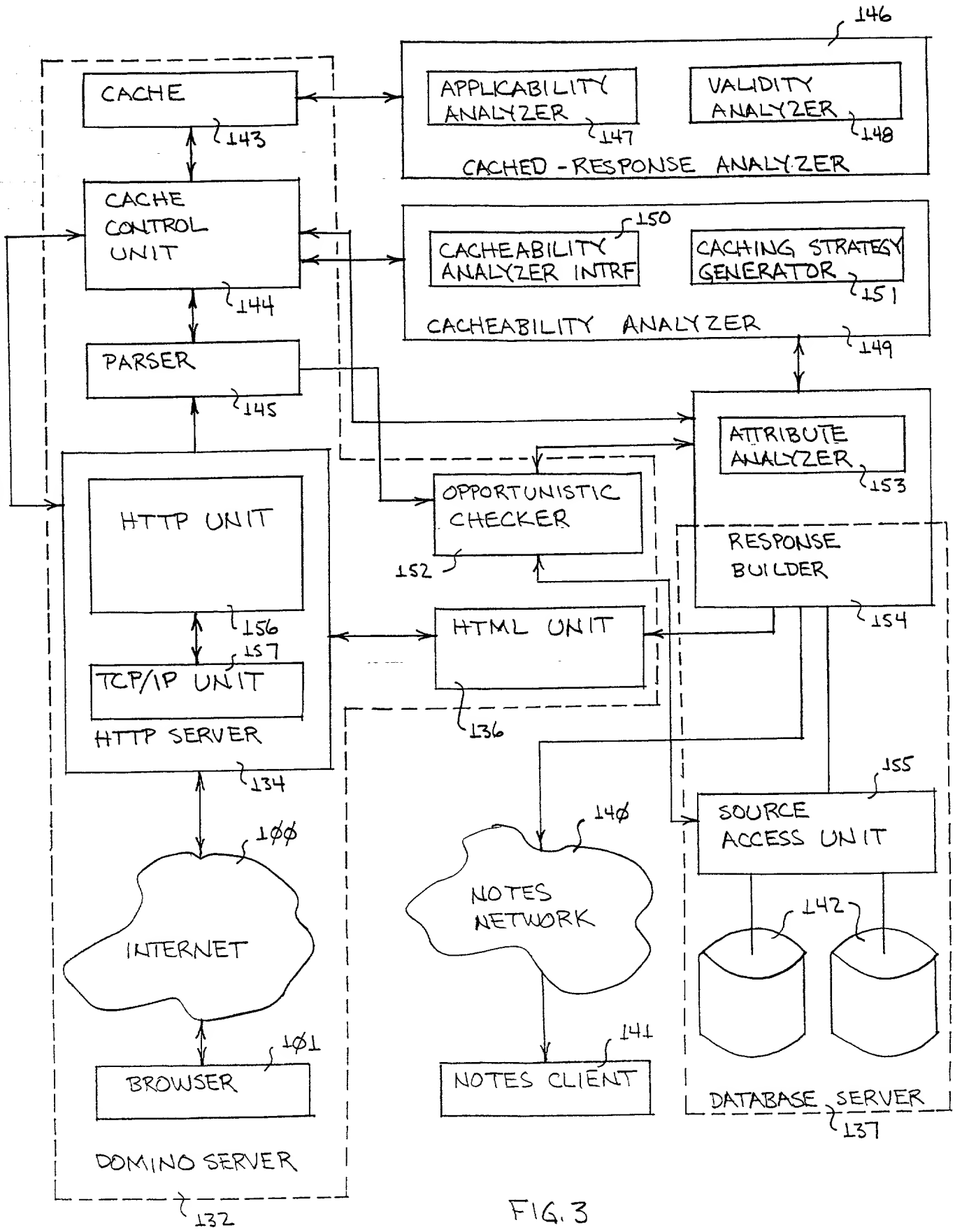

146
CACHE
143
APPLICABILITY ANALYZER
147
VALIDITY ANALYZER
148
CACHED - RESPONSE ANALYZER
CACHE CONTROL UNIT
144
150
CACHEABILITY ANALYZER INTRF
CACHING STRATEGY GENERATOR
151
CACHEABILITY ANALYZER
149
PARSER
145
ATTRIBUTE ANALYZER
153
HTTP UNIT
OPPORTUNISTIC CHECKER
152
RESPONSE BUILDER
154
156
157
TCP/IP UNIT
HTTP SERVER
HTML UNIT
136
134
155
SOURCE ACCESS UNIT
140
100
NOTES NETWORK
INTERNET
142
101
141
BROWSER
NOTES CLIENT
DATABASE SERVER
137
DOMINO SERVER
132

FIG. 3

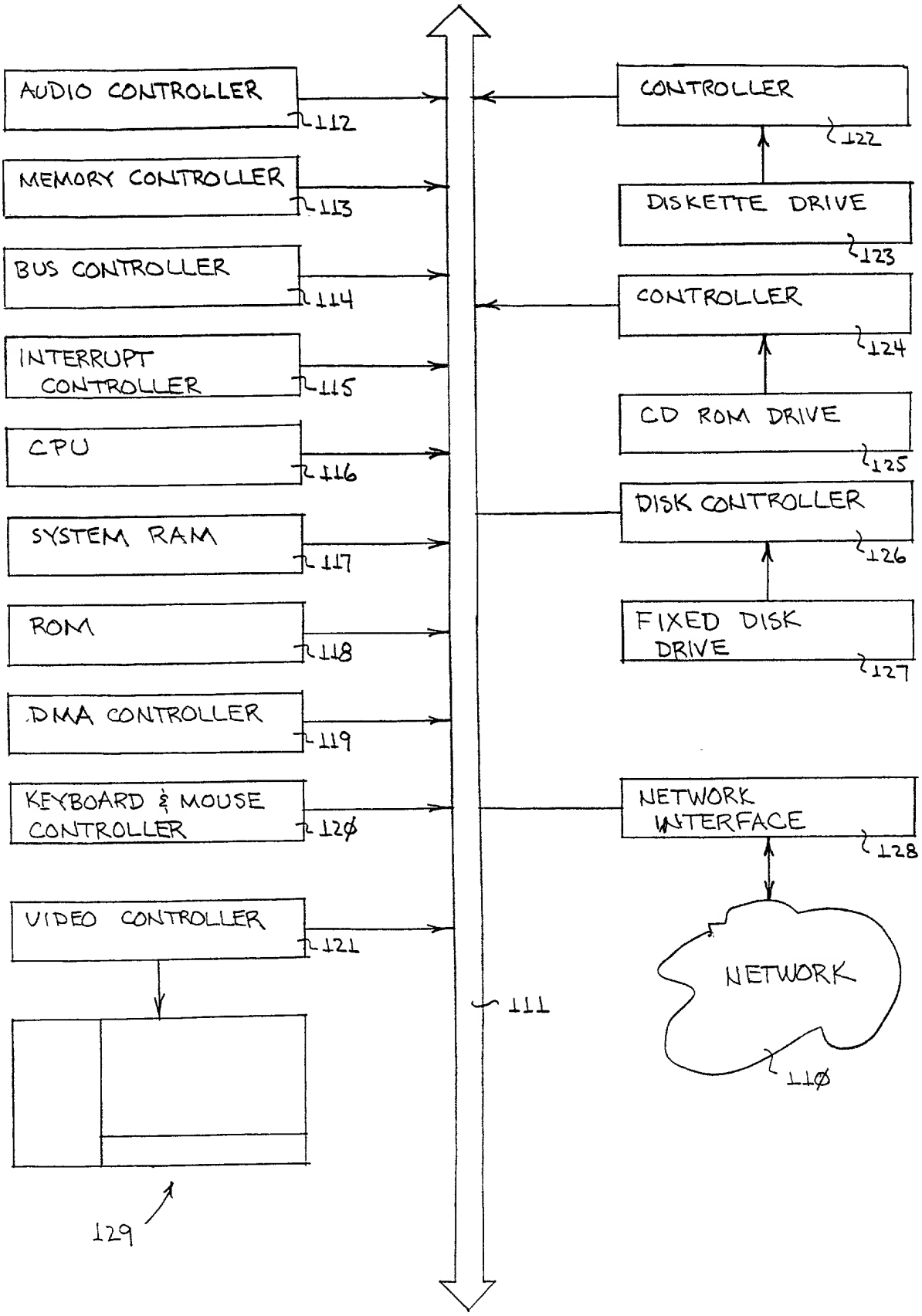


FIG. 4

PRIOR ART

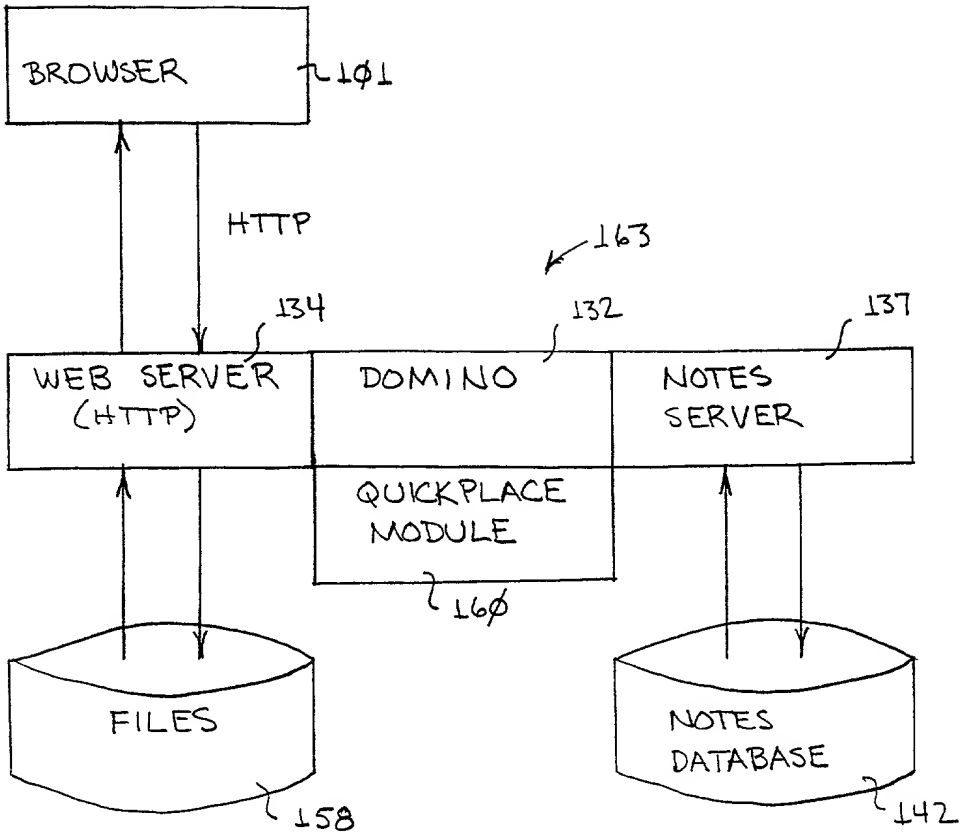

BROWSER
101
HTTP
163
134
132
137
WEB SERVER
(HTTP)
DOMINO
NOTES
SERVER
QUICKPLACE
MODULE
160
FILES
158
NOTES
DATABASE
142

FIG. 5

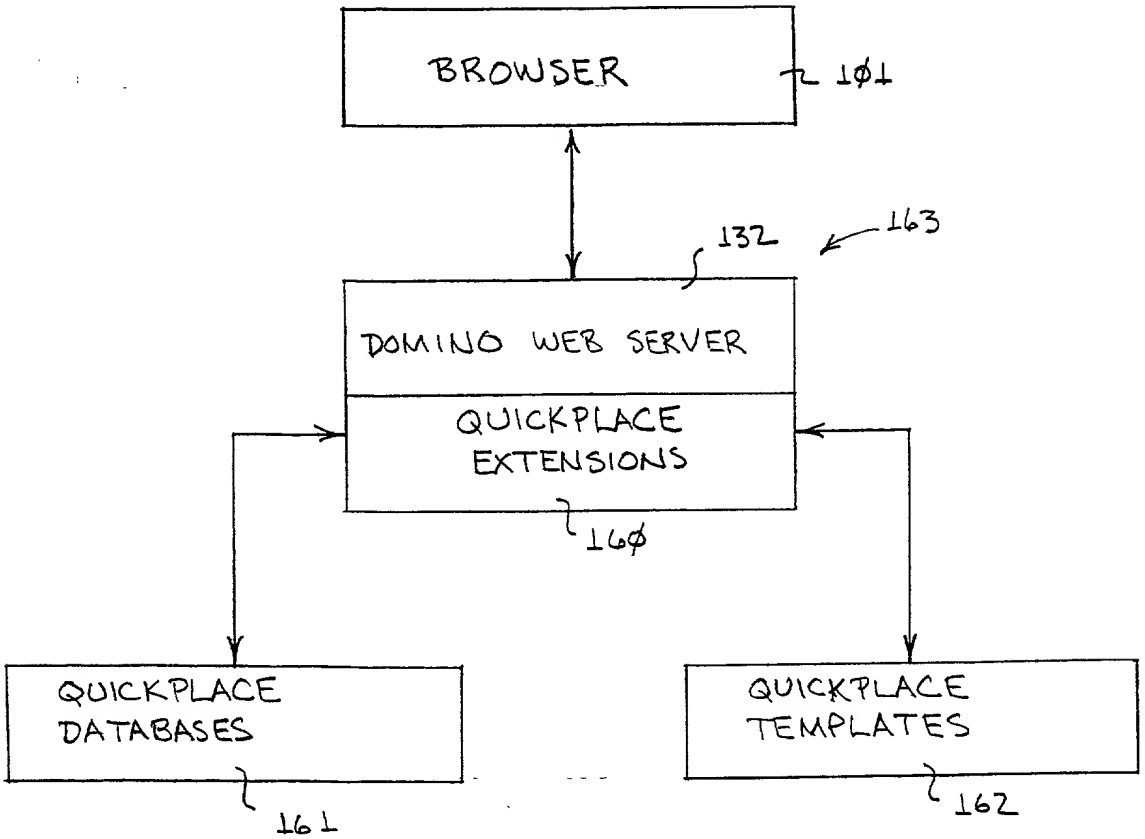

BROWSER
101
132
163
DOMINO WEB SERVER
QUICKPLACE EXTENSIONS
160
QUICKPLACE DATABASES
161
QUICKPLACE TEMPLATES
162

FIG. 6

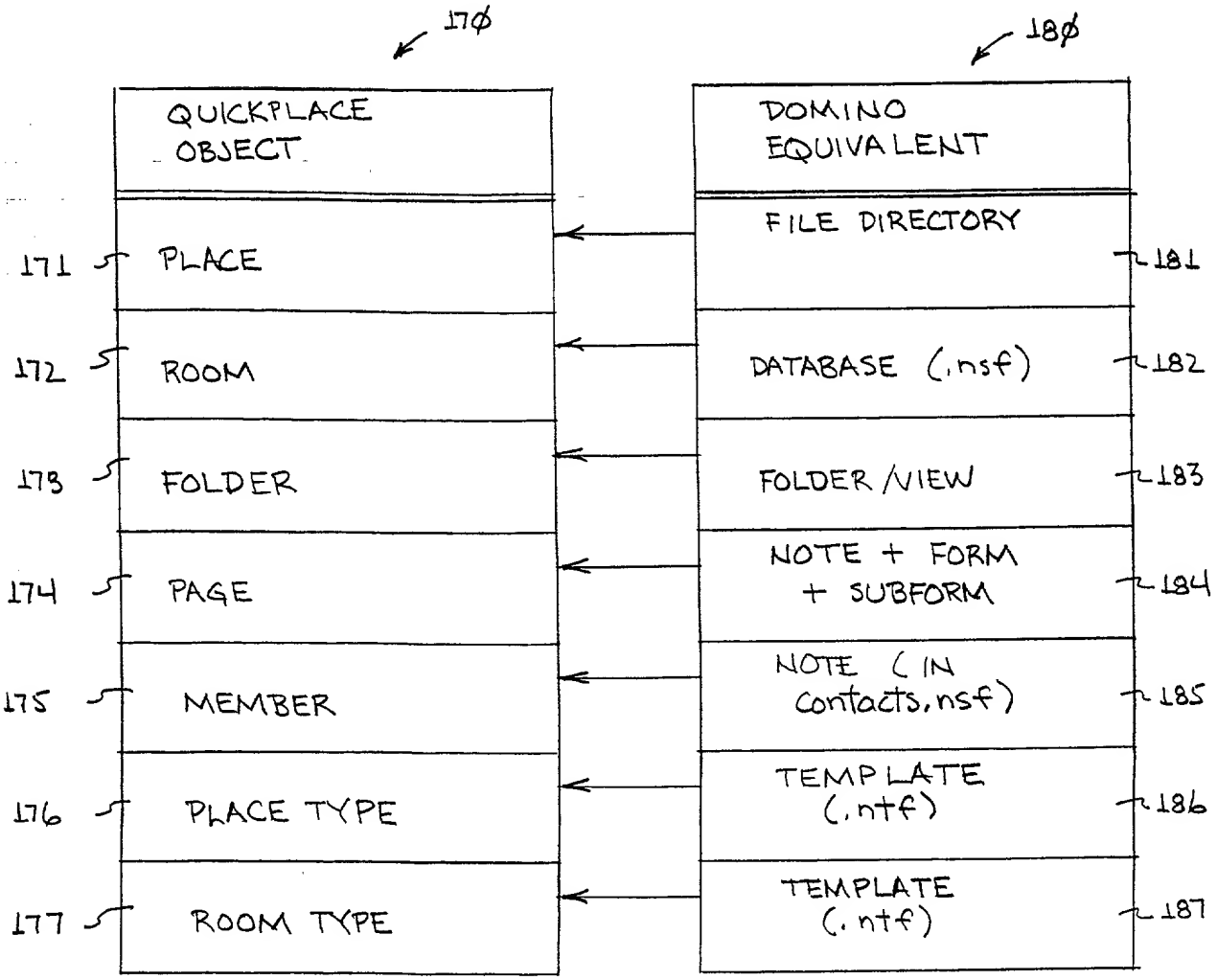

170
180
QUICKPLACE OBJECT
DOMINO EQUIVALENT
171
PLACE
FILE DIRECTORY
181
172
ROOM
DATABASE (.nsf)
182
173
FOLDER
FOLDER/VIEW
183
174
PAGE
NOTE + FORM + SUBFORM
184
175
MEMBER
NOTE (IN Contacts.nsf)
185
176
PLACE TYPE
TEMPLATE (.ntf)
186
177
ROOM TYPE
TEMPLATE (.ntf)
187

FIG. 7

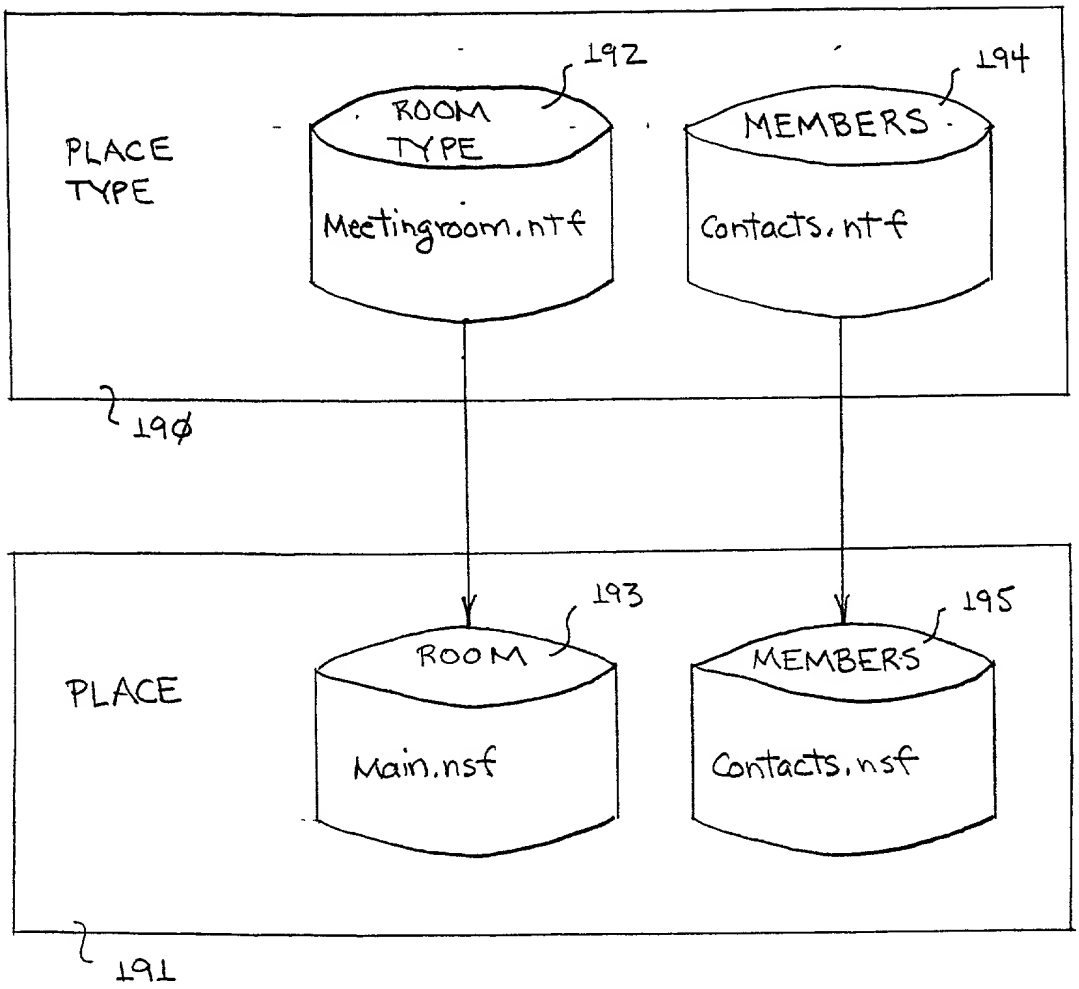

PLACE TYPE
ROOM TYPE
192
MEMBERS
194
Meetingroom.ntf
Contacts.ntf
190
PLACE
ROOM
193
MEMBERS
195
Main.nsf
Contacts.nsf
191

FIG. 8

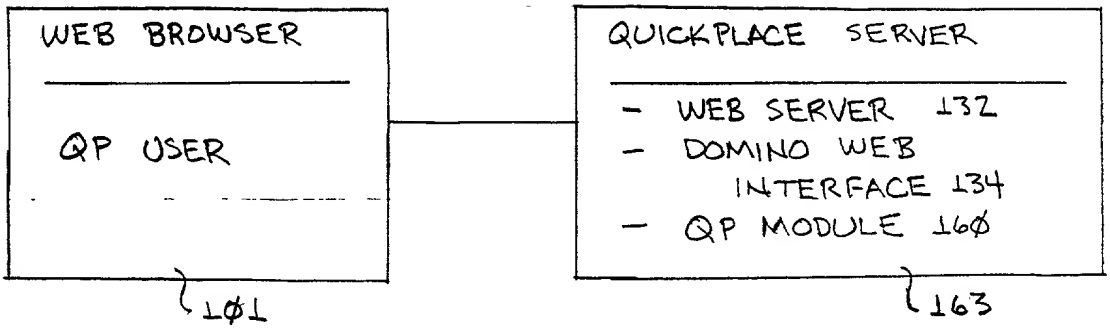

WEB BROWSER
QP USER
101
QUICKPLACE SERVER
- WEB SERVER 132
- DOMINO WEB INTERFACE 134
- QP MODULE 160
163

FIG. 9

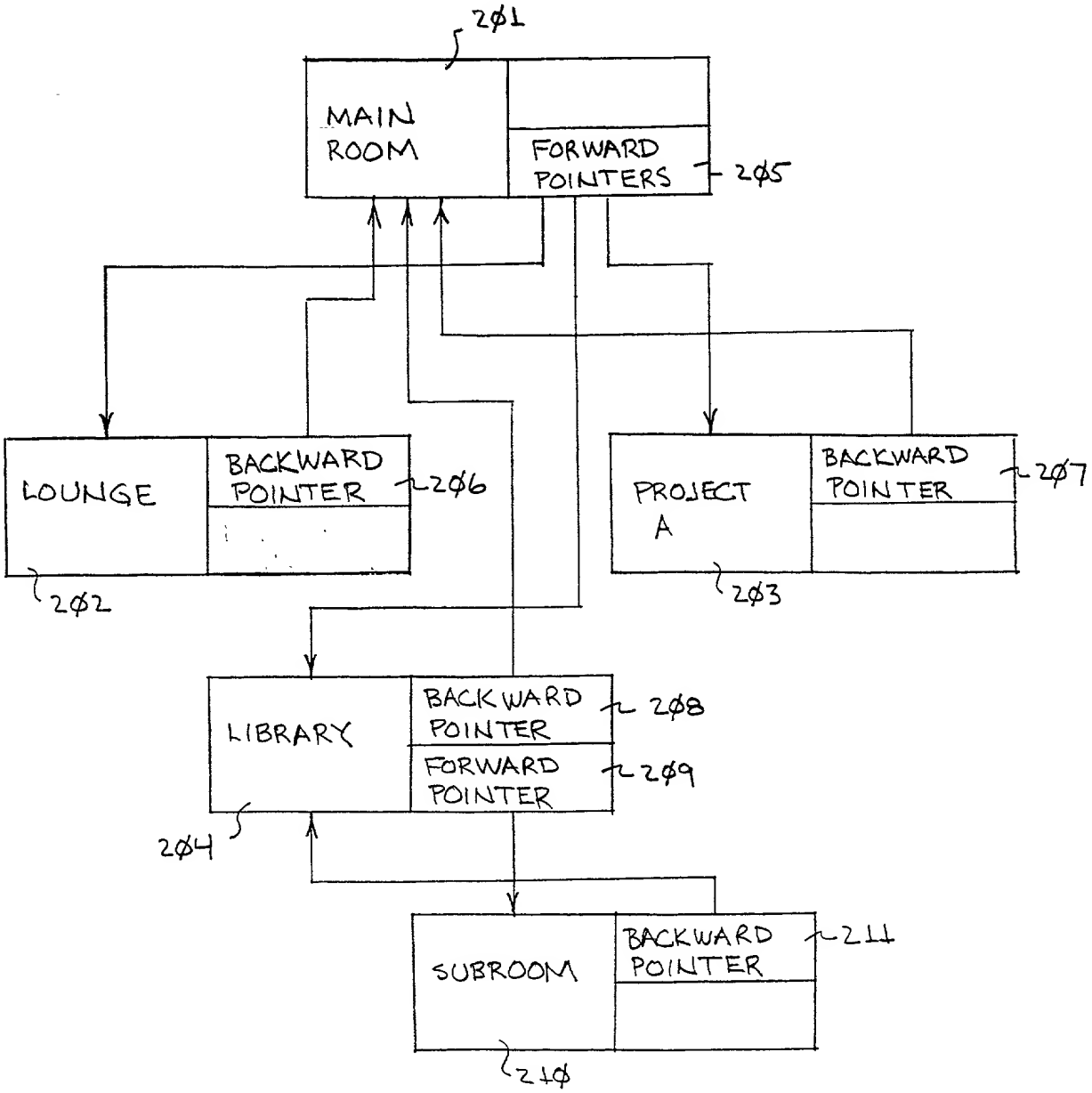

201
MAIN ROOM
FORWARD POINTERS
205
LOUNGE
BACKWARD POINTER
206
202
PROJECT A
BACKWARD POINTER
207
203
LIBRARY
BACKWARD POINTER
208
FORWARD POINTER
209
204
SUBROOM
BACKWARD POINTER
211
210

FIG. 10

FORWARD POINTER 209

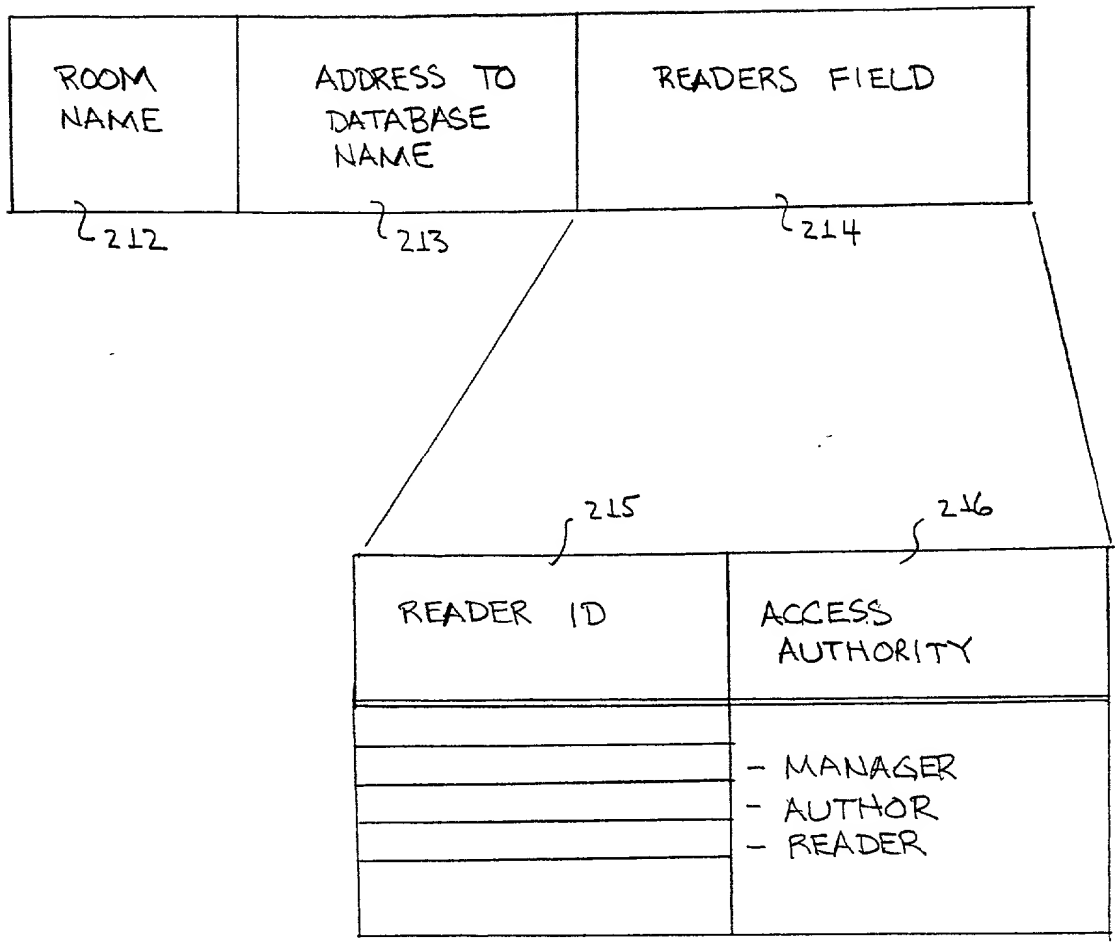


FIG. 11

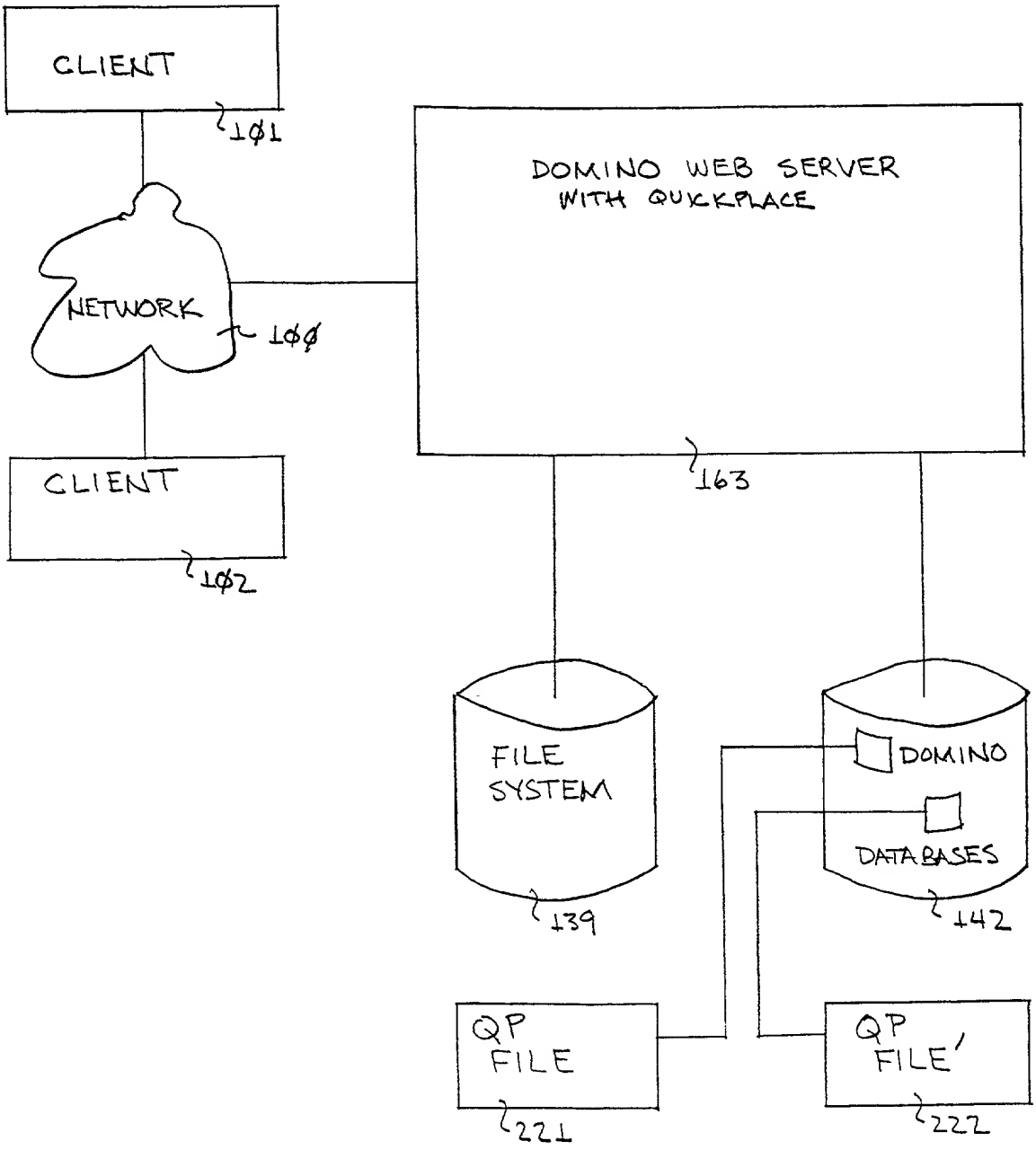

CLIENT
101
DOMINO WEB SERVER
WITH QUICKPLACE
NETWORK
100
CLIENT
102
163
FILE
SYSTEM
139
DOMINO
DATABASES
142
QP
FILE
221
QP
FILE'
222

FIG. 12

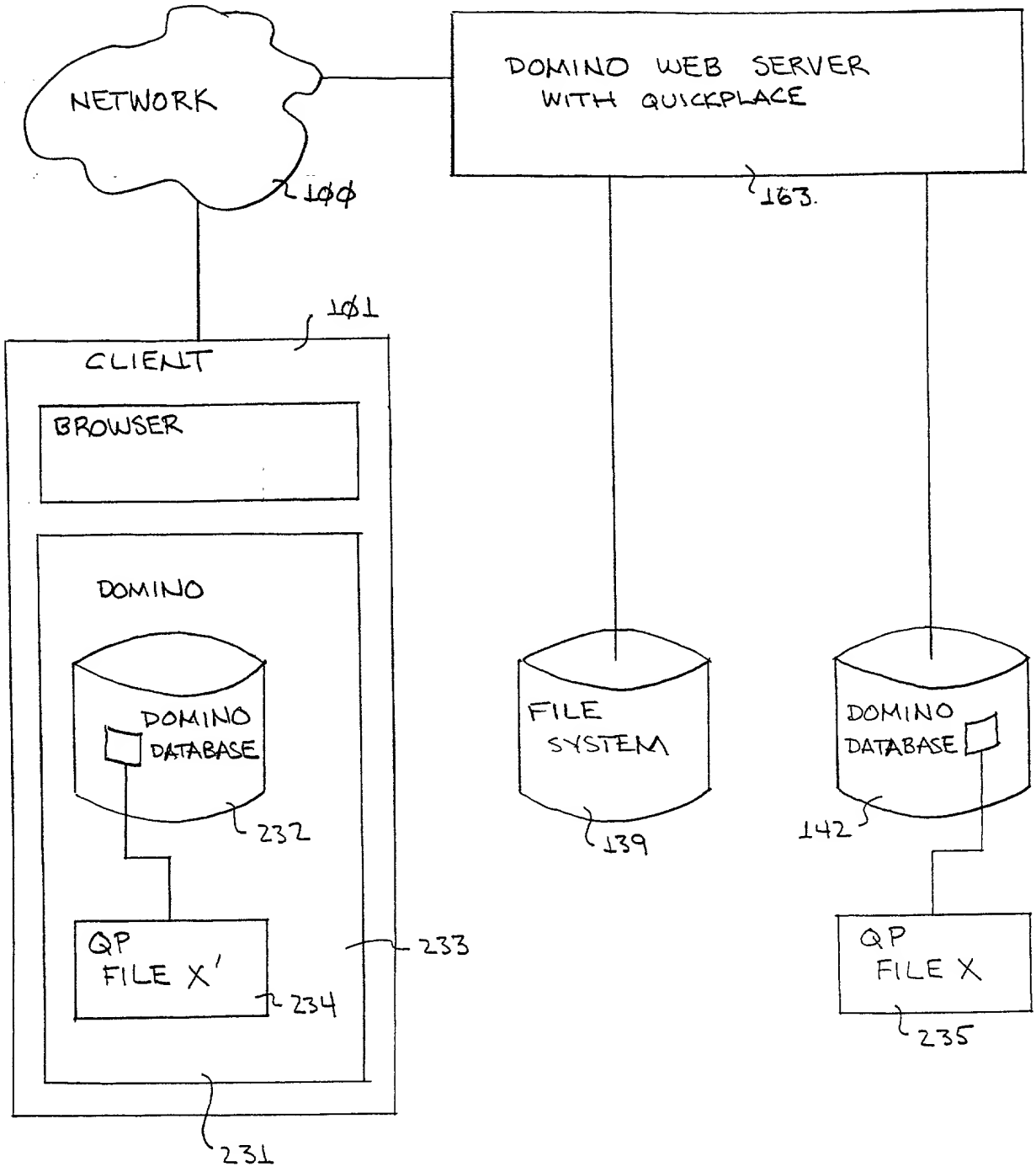

NETWORK
100
DOMINO WEB SERVER
WITH QUICKPLACE
163
101
CLIENT
BROWSER
DOMINO
DOMINO
DATABASE
232
QP
FILE X'
234
233
231
FILE
SYSTEM
139
DOMINO
DATABASE
142
QP
FILE X
235

FIG. 13

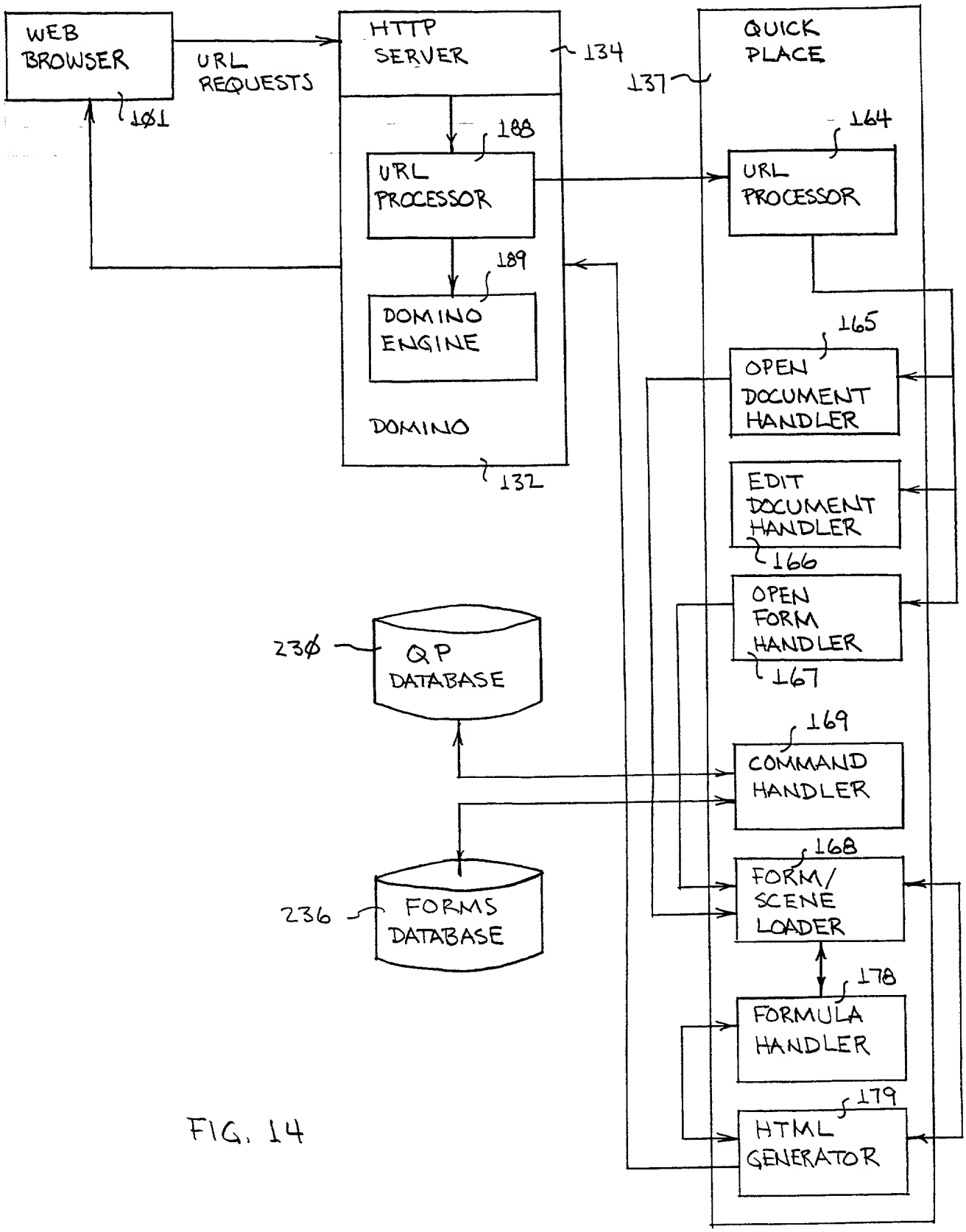

WEB BROWSER
101
URL REQUESTS
HTTP SERVER
134
188
URL PROCESSOR
189
DOMINO ENGINE
DOMINO
132
QUICK PLACE
137
164
URL PROCESSOR
165
OPEN DOCUMENT HANDLER
EDIT DOCUMENT HANDLER
166
OPEN FORM HANDLER
167
230
QP DATABASE
169
COMMAND HANDLER
168
FORM/ SCENE LOADER
178
FORMULA HANDLER
179
HTML GENERATOR
236
FORMS DATABASE

FIG. 14

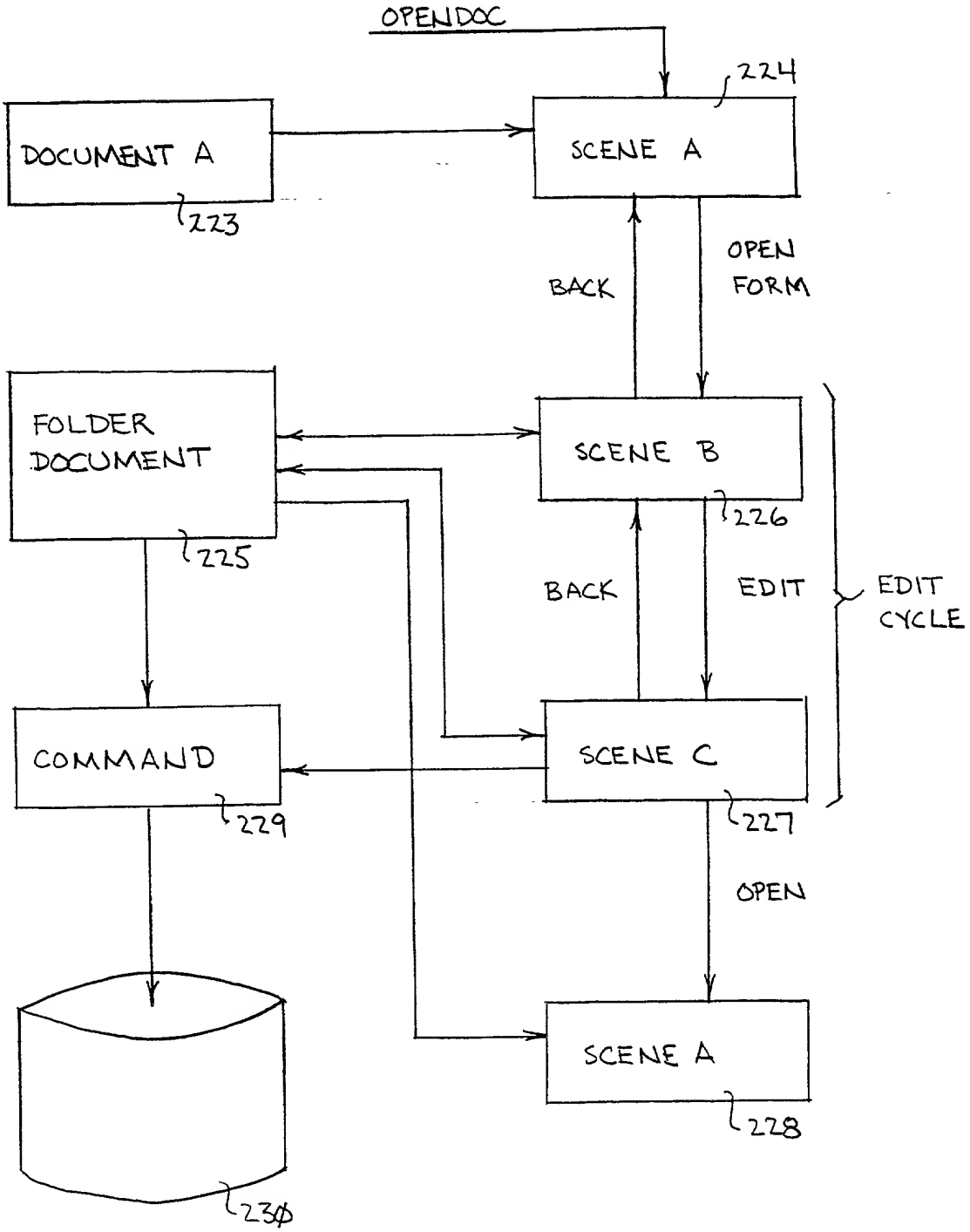

OPENDOC
224
DOCUMENT A
223
SCENE A
BACK
OPEN
FORM
FOLDER
DOCUMENT
225
SCENE B
226
BACK
EDIT
EDIT
CYCLE
COMMAND
229
SCENE C
227
OPEN
SCENE A
228
230

FIG. 15

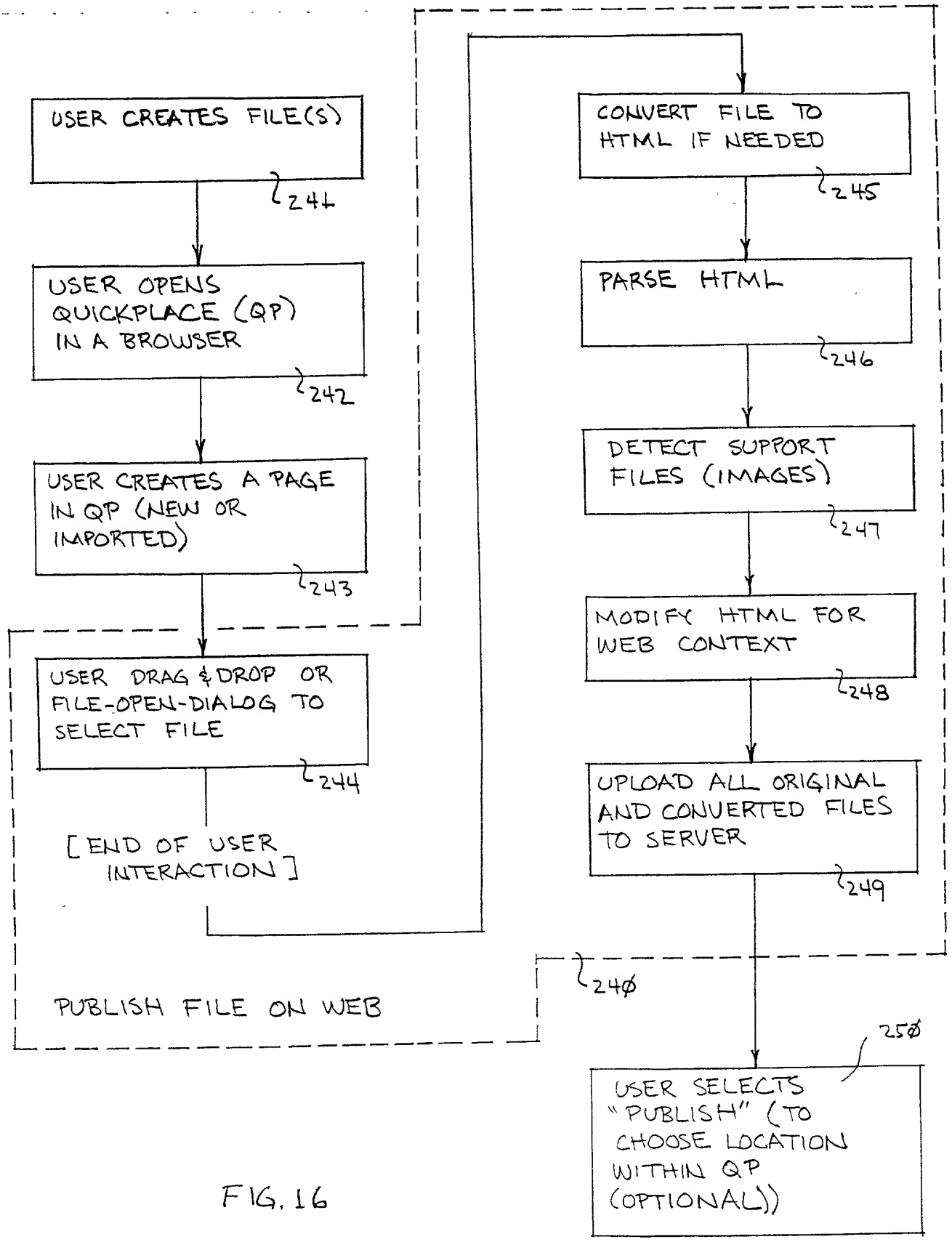

USER CREATES FILE(S)
241
USER OPENS QUICKPLACE (QP) IN A BROWSER
242
USER CREATES A PAGE IN QP (NEW OR IMPORTED)
243
USER DRAG & DROP OR FILE-OPEN-DIALOG TO SELECT FILE
244
[END OF USER INTERACTION]
CONVERT FILE TO HTML IF NEEDED
245
PARSE HTML
246
DETECT SUPPORT FILES (IMAGES)
247
MODIFY HTML FOR WEB CONTEXT
248
UPLOAD ALL ORIGINAL AND CONVERTED FILES TO SERVER
249
240
PUBLISH FILE ON WEB
250
USER SELECTS "PUBLISH" (TO CHOOSE LOCATION WITHIN QP (OPTIONAL))

FIG. 16

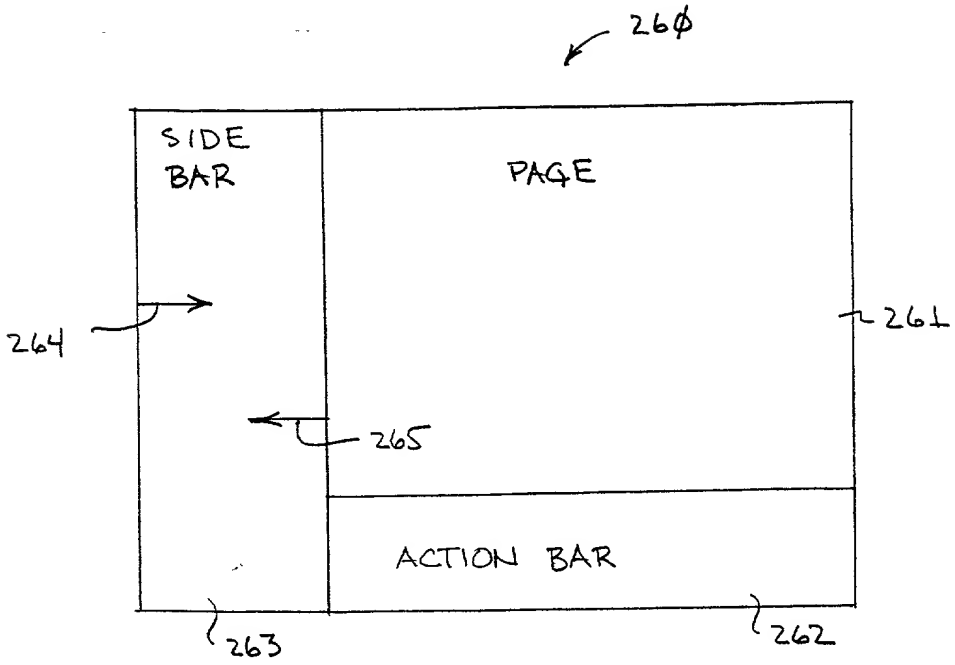


FIG. 17

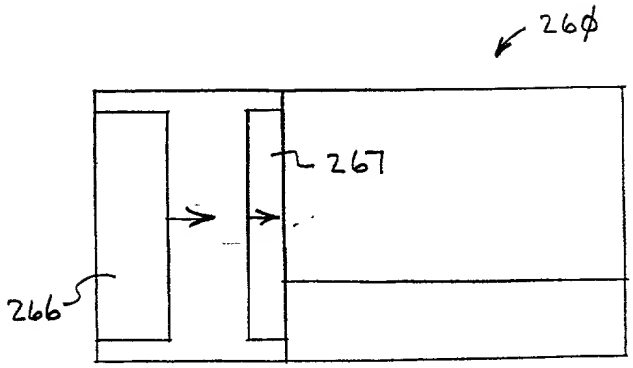


FIG. 18

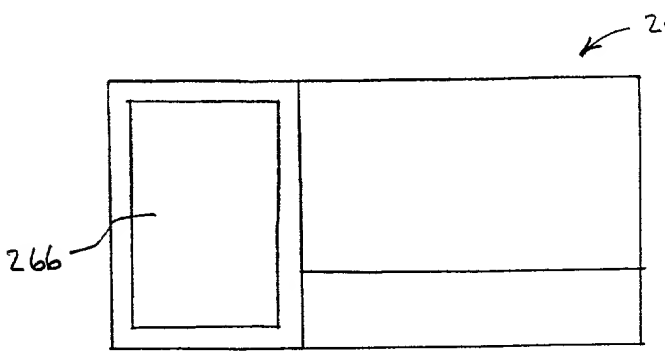


FIG. 19

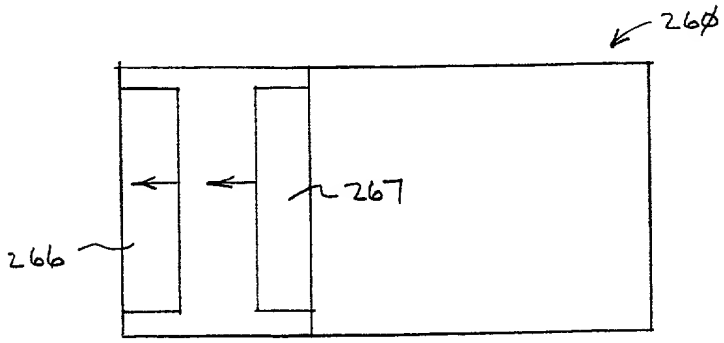


FIG. 20

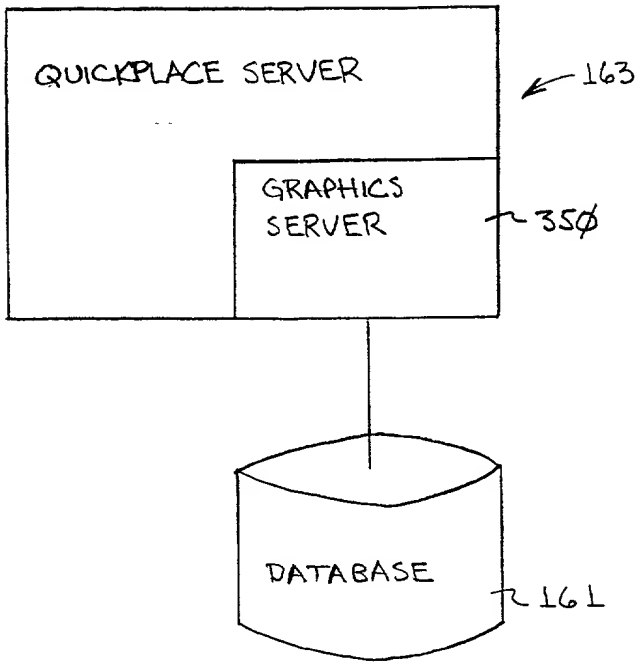


FIG. 21

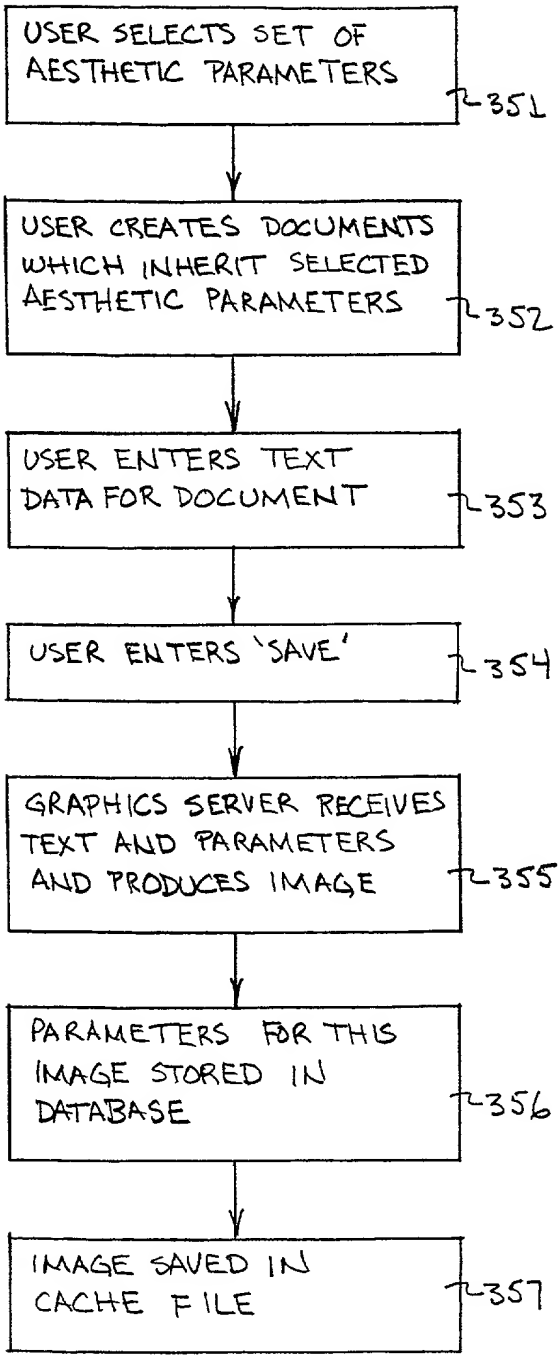

USER SELECTS SET OF AESTHETIC PARAMETERS
351
USER CREATES DOCUMENTS WHICH INHERIT SELECTED AESTHETIC PARAMETERS
352
USER ENTERS TEXT DATA FOR DOCUMENT
353
USER ENTERS 'SAVE'
354
GRAPHICS SERVER RECEIVES TEXT AND PARAMETERS AND PRODUCES IMAGE
355
PARAMETERS FOR THIS IMAGE STORED IN DATABASE
356
IMAGE SAVED IN CACHE FILE
357

FIG. 22

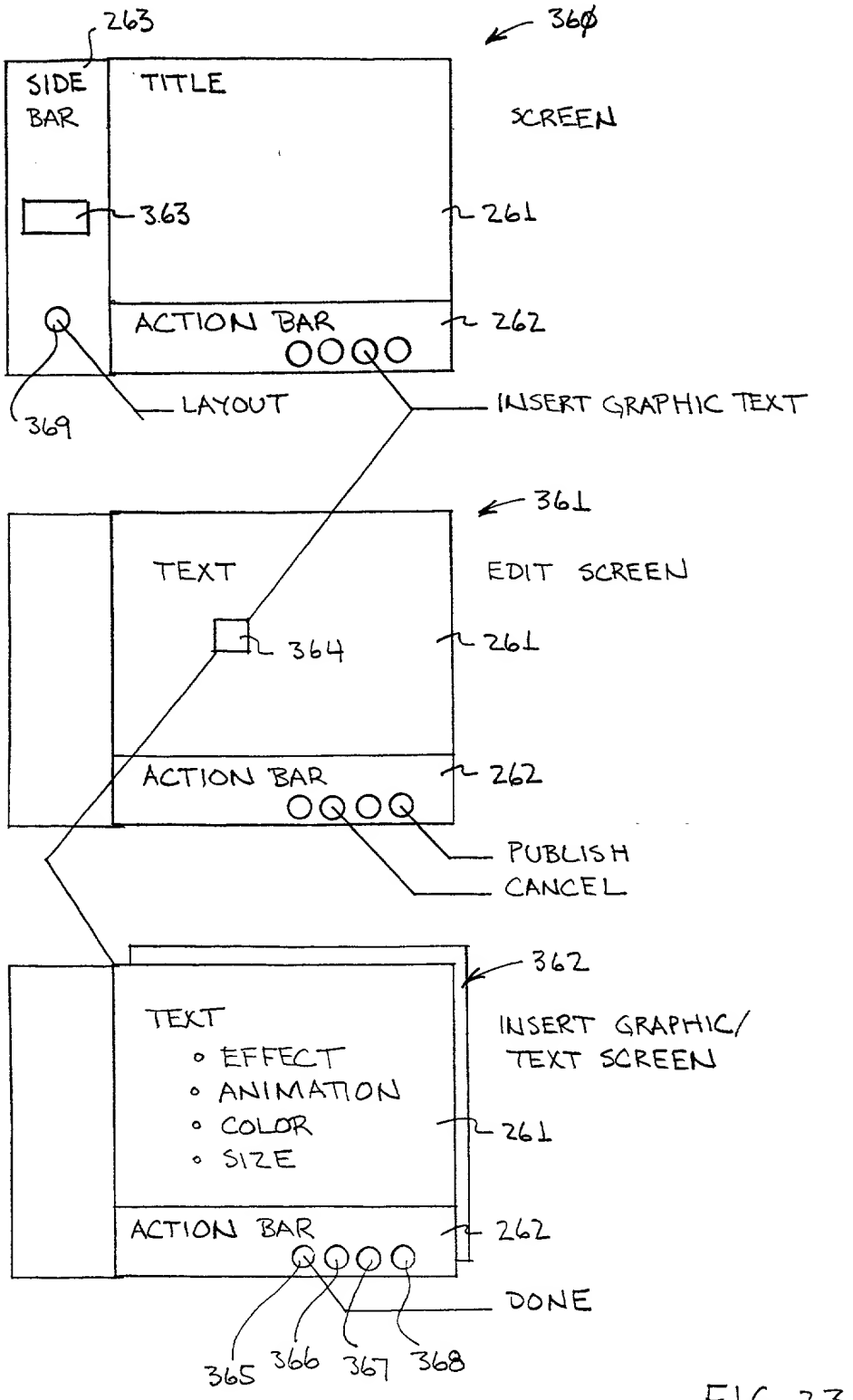

263
360
SIDE BAR
TITLE
SCREEN
363
261
ACTION BAR
262
369
LAYOUT
INSERT GRAPHIC TEXT
361
TEXT
EDIT SCREEN
364
261
ACTION BAR
262
PUBLISH
CANCEL
362
TEXT
EFFECT
ANIMATION
COLOR
SIZE
INSERT GRAPHIC/ TEXT SCREEN
261
ACTION BAR
262
DONE
365
366
367
368

FIG. 23

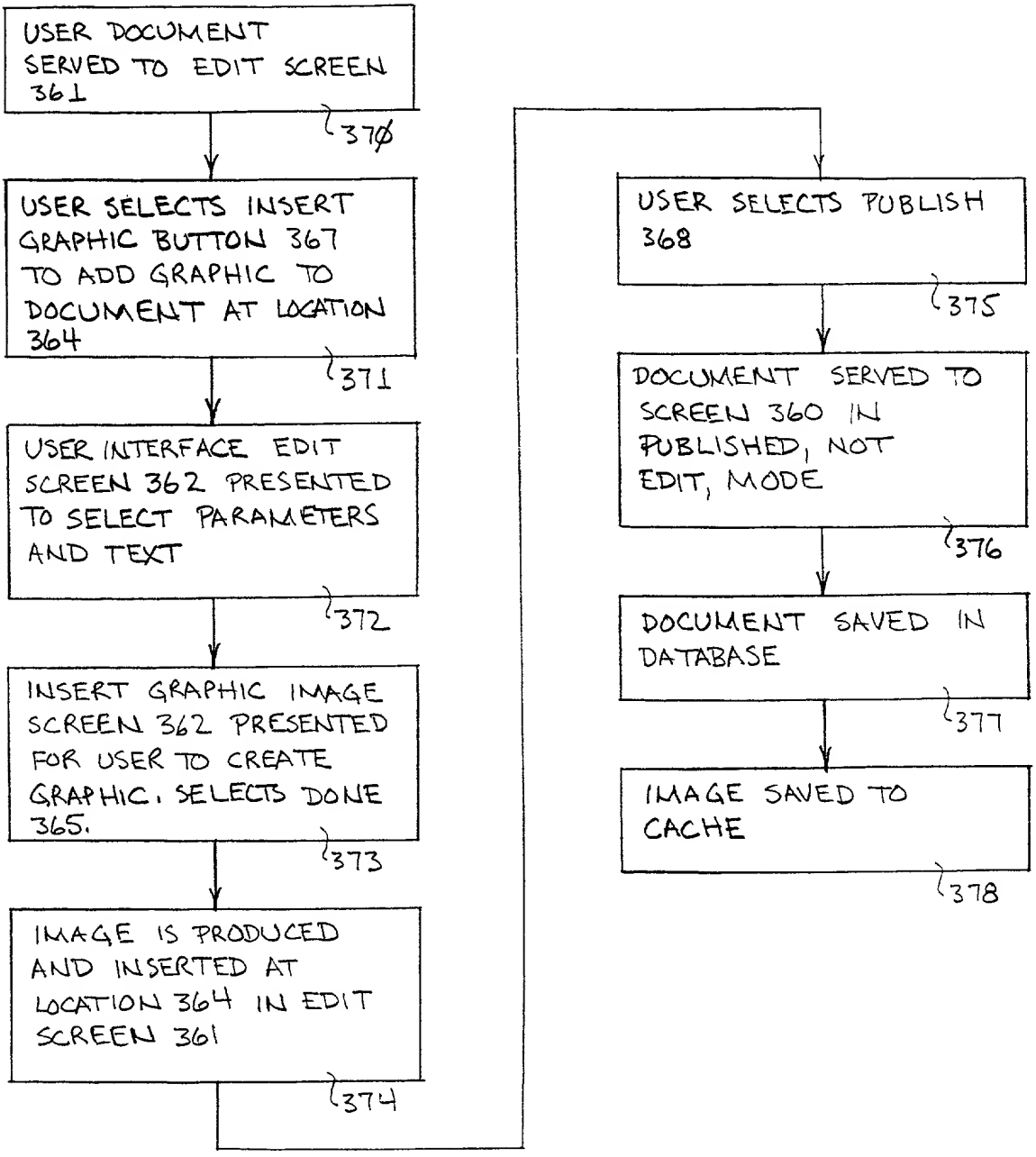

USER DOCUMENT SERVED TO EDIT SCREEN 361
370
USER SELECTS INSERT GRAPHIC BUTTON 367 TO ADD GRAPHIC TO DOCUMENT AT LOCATION 364
371
USER INTERFACE EDIT SCREEN 362 PRESENTED TO SELECT PARAMETERS AND TEXT
372
INSERT GRAPHIC IMAGE SCREEN 362 PRESENTED FOR USER TO CREATE GRAPHIC. SELECTS DONE 365.
373
IMAGE IS PRODUCED AND INSERTED AT LOCATION 364 IN EDIT SCREEN 361
374
USER SELECTS PUBLISH 368
375
DOCUMENT SERVED TO SCREEN 360 IN PUBLISHED, NOT EDIT, MODE
376
DOCUMENT SAVED IN DATABASE
377
IMAGE SAVED TO CACHE
378

FIG. 24

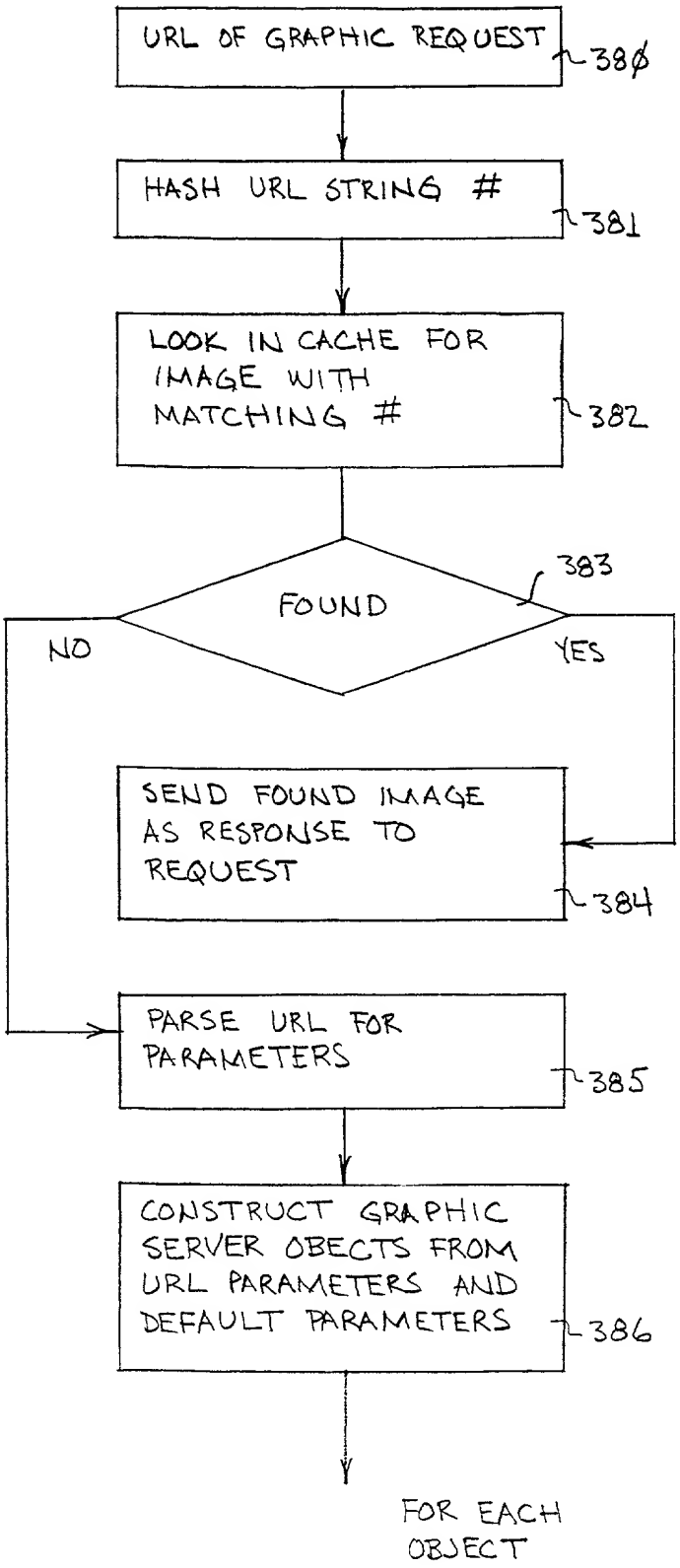

URL OF GRAPHIC REQUEST
380
HASH URL STRING #
381
LOOK IN CACHE FOR IMAGE WITH MATCHING #
382
FOUND
383
NO
YES
SEND FOUND IMAGE AS RESPONSE TO REQUEST
384
PARSE URL FOR PARAMETERS
385
CONSTRUCT GRAPHIC SERVER OBECTS FROM URL PARAMETERS AND DEFAULT PARAMETERS
386
FOR EACH OBJECT

FIG. 25

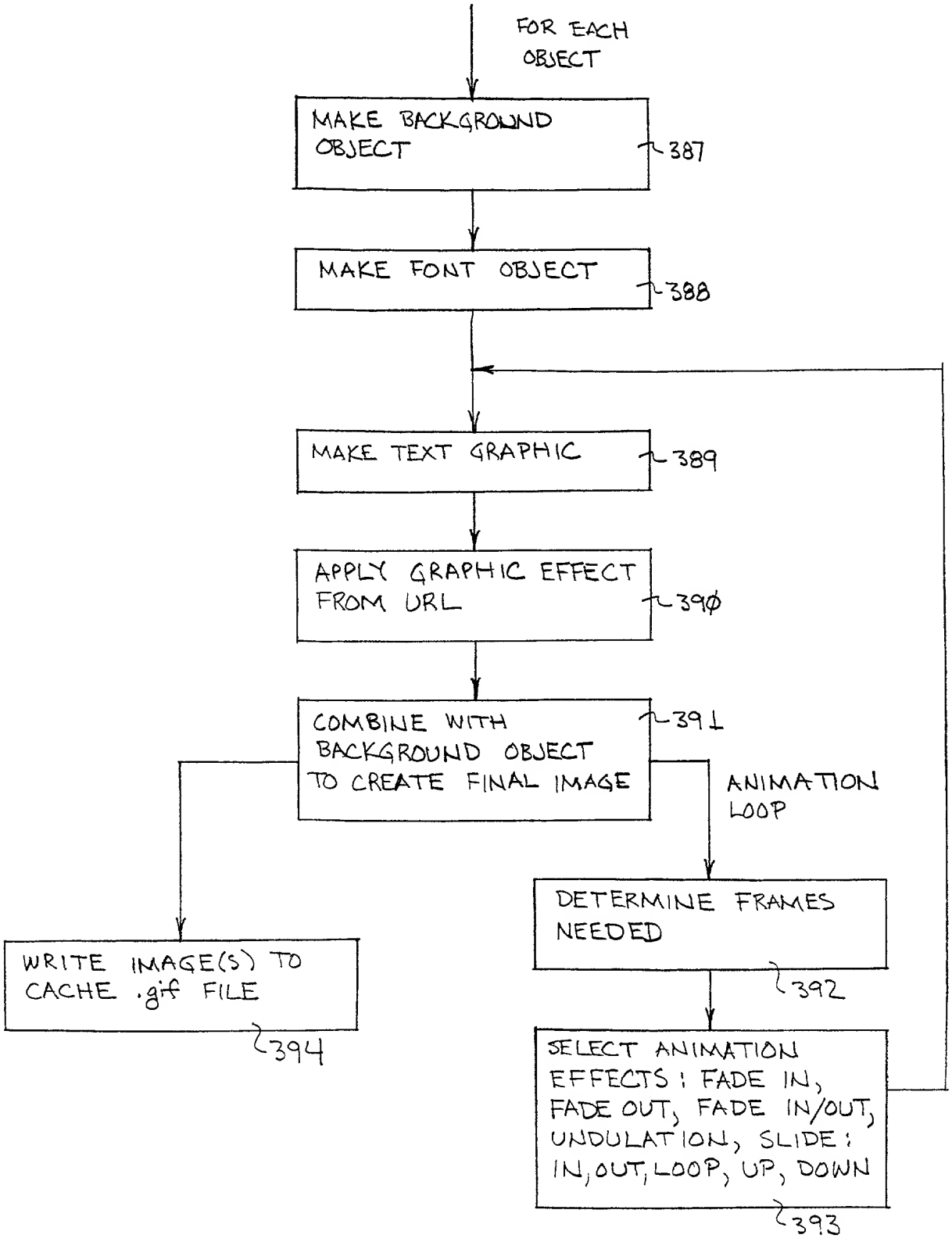

FOR EACH OBJECT
MAKE BACKGROUND OBJECT
387
MAKE FONT OBJECT
388
MAKE TEXT GRAPHIC
389
APPLY GRAPHIC EFFECT FROM URL
390
COMBINE WITH BACKGROUND OBJECT TO CREATE FINAL IMAGE
391
ANIMATION LOOP
DETERMINE FRAMES NEEDED
392
SELECT ANIMATION EFFECTS : FADE IN, FADE OUT, FADE IN/OUT, UNDULATION, SLIDE : IN, OUT, LOOP, UP, DOWN
393
WRITE IMAGE(S) TO CACHE .gif FILE
394

FIG. 26

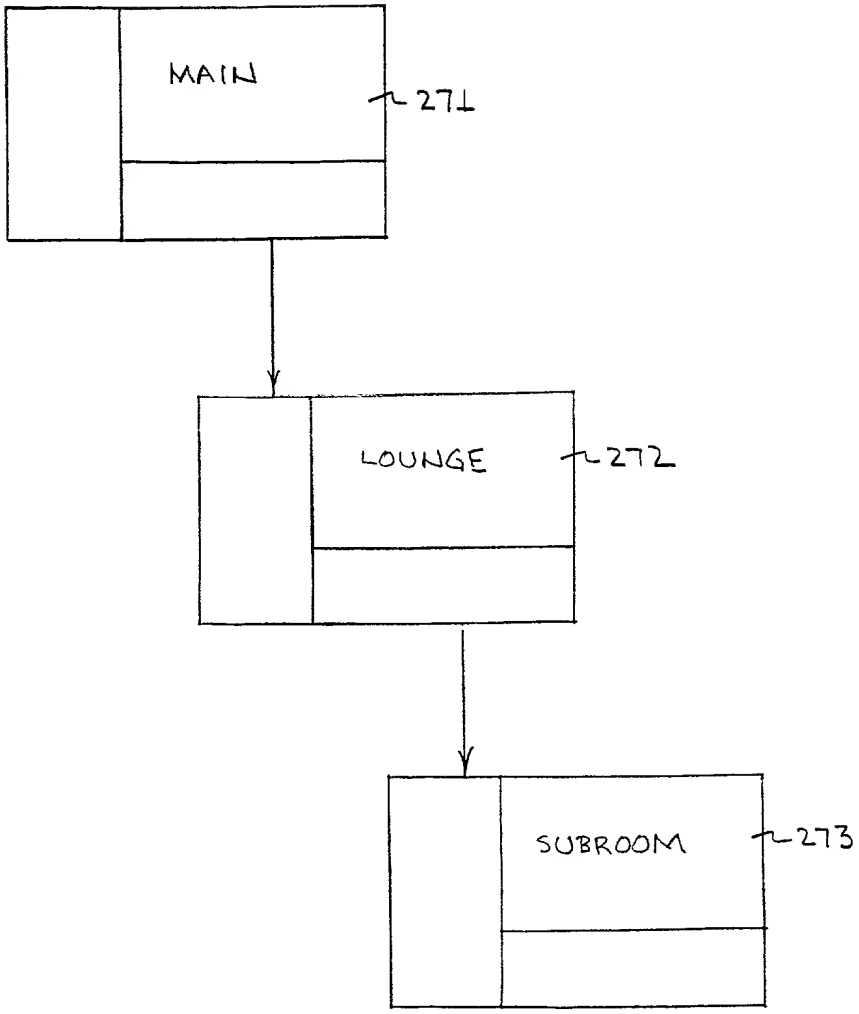


FIG. 27

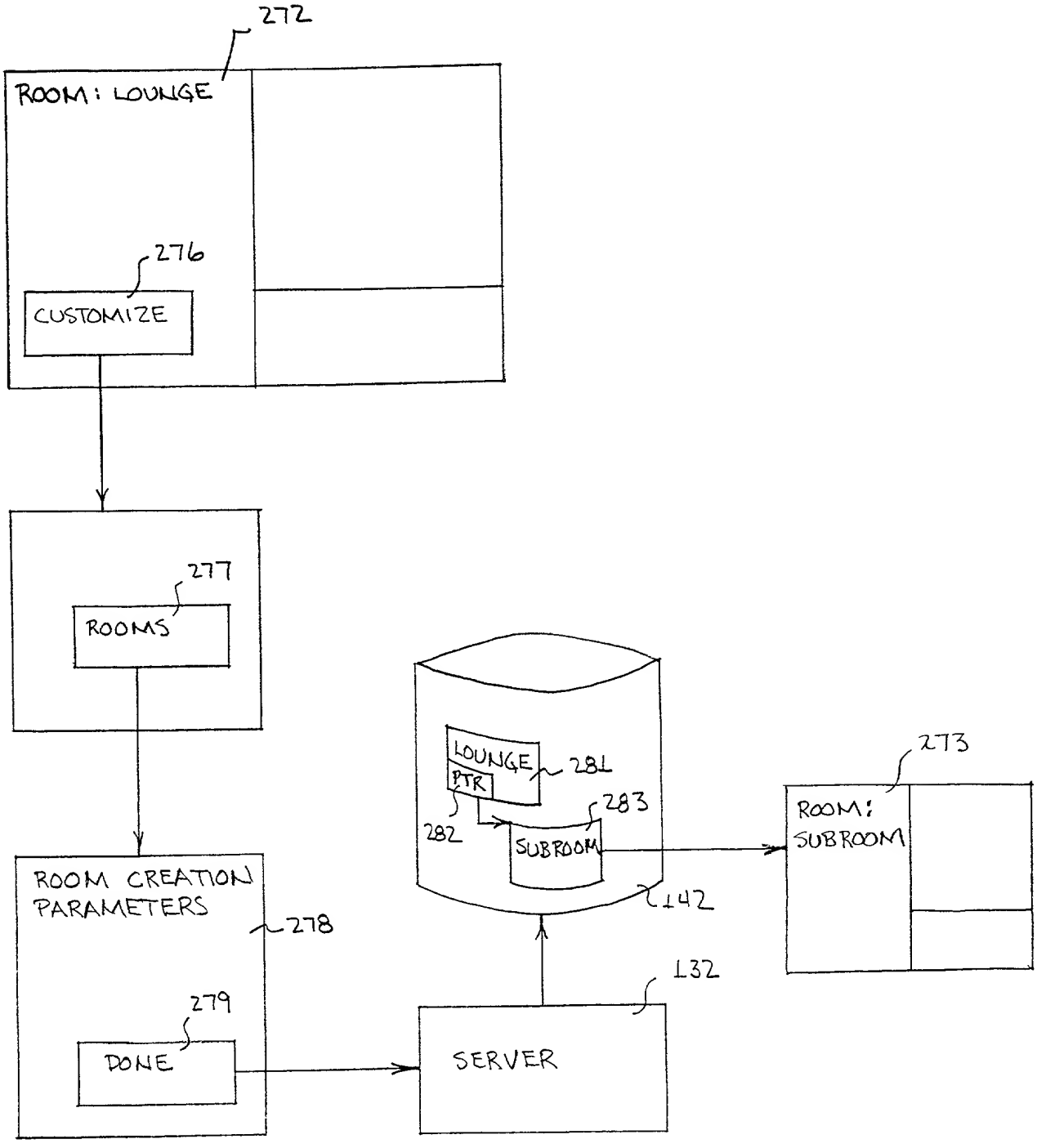

272
ROOM: LOUNGE
276
CUSTOMIZE
277
ROOMS
ROOM CREATION PARAMETERS
278
279
DONE
LOUNGE
PTR
281
282
283
SUBROOM
142
273
ROOM: SUBROOM
132
SERVER

FIG. 28

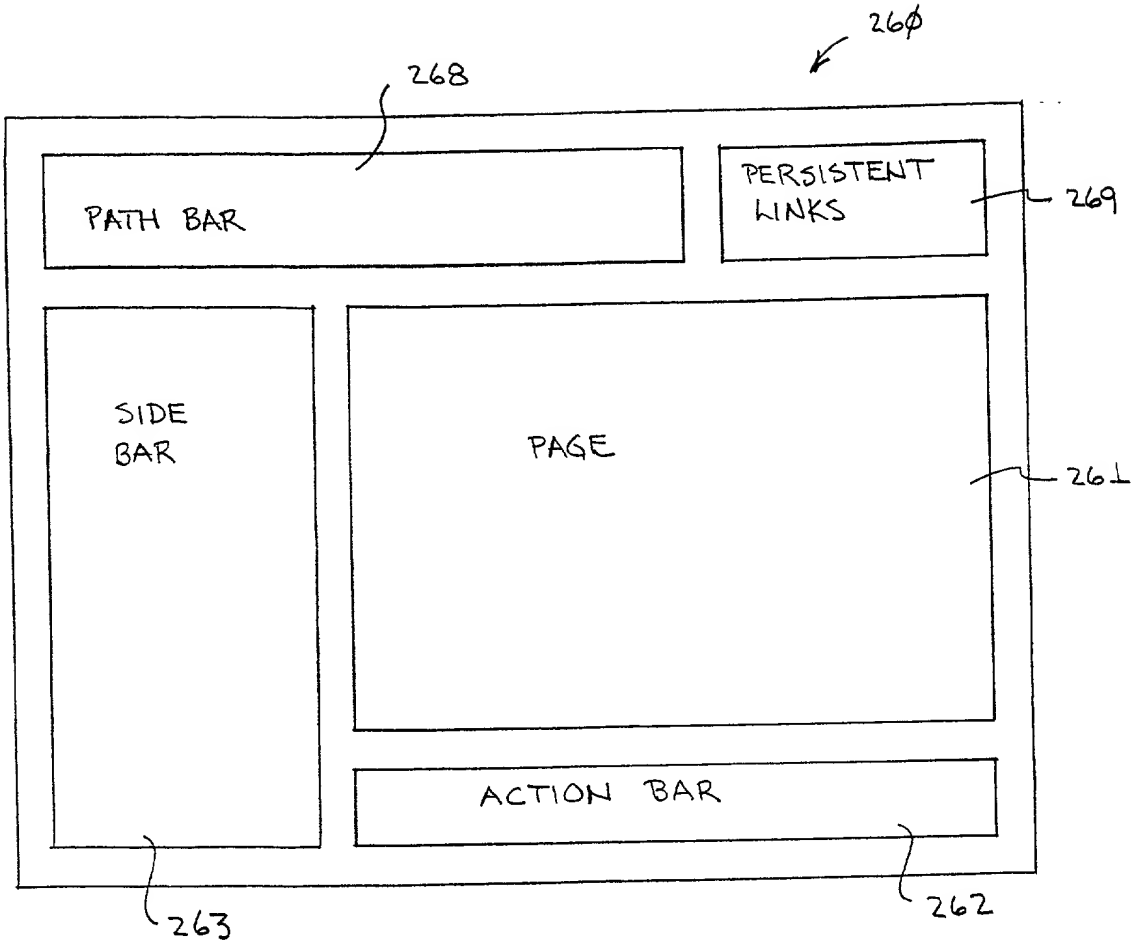

260
268
PATH BAR
PERSISTENT LINKS
269
SIDE BAR
PAGE
261
ACTION BAR
263
262

FIG. 29

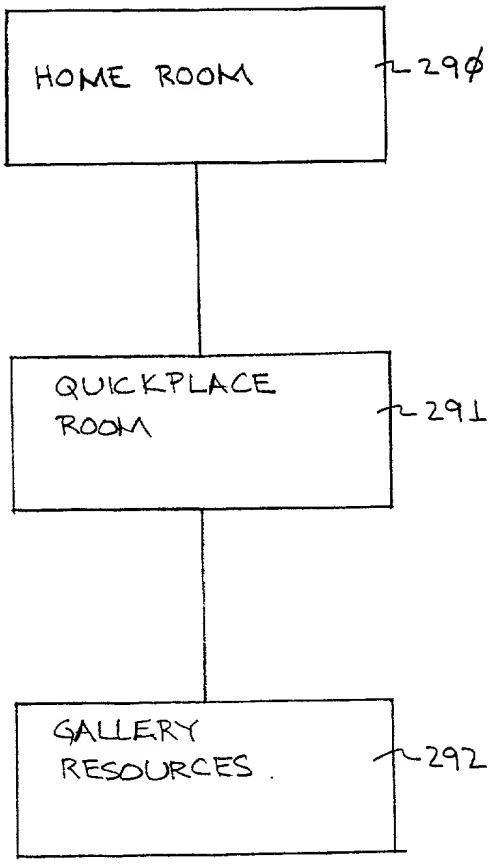


FIG. 30

Copyright © 2006 IBM

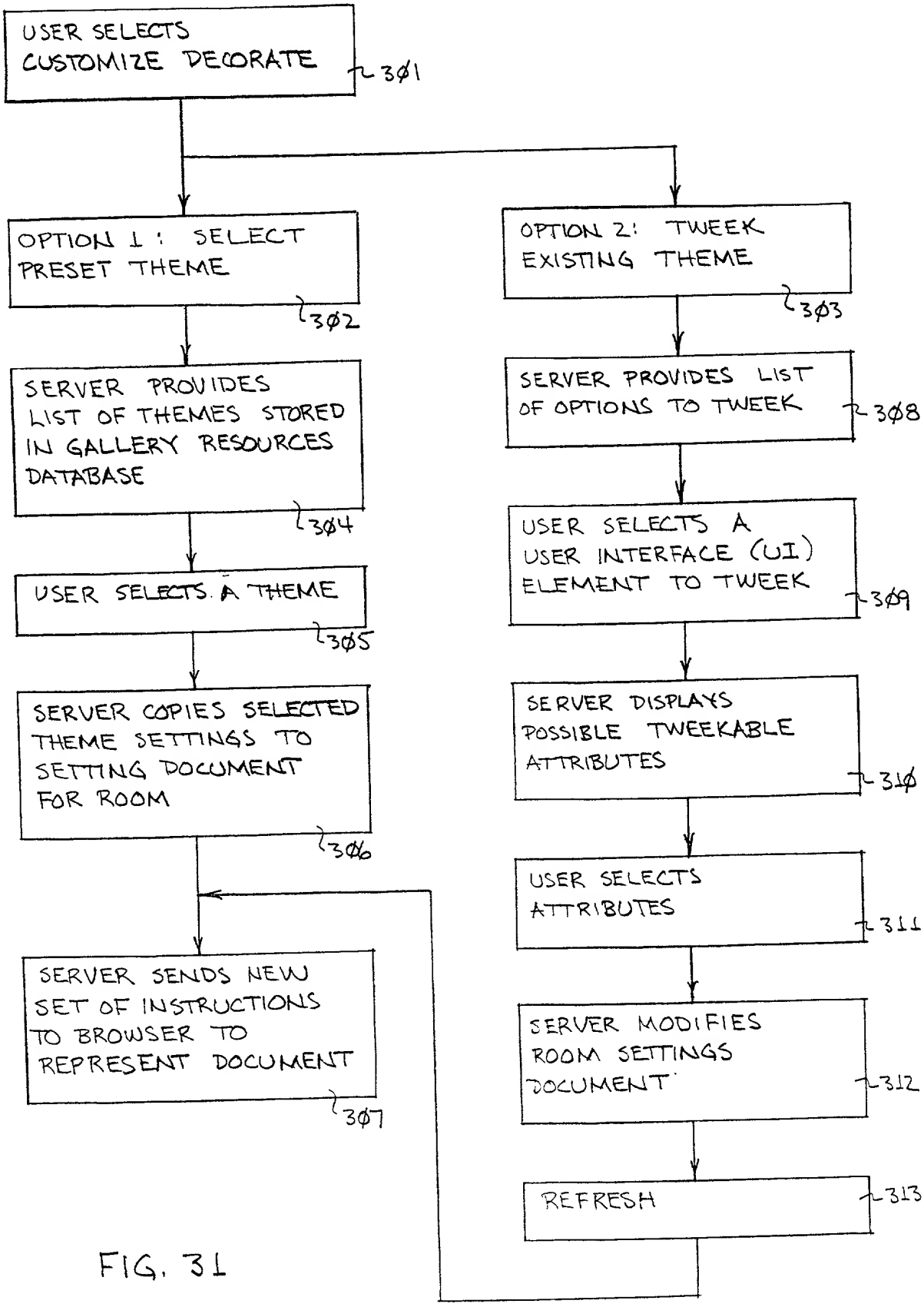

USER SELECTS CUSTOMIZE DECORATE
301
OPTION 1: SELECT PRESET THEME
302
SERVER PROVIDES LIST OF THEMES STORED IN GALLERY RESOURCES DATABASE
304
USER SELECTS A THEME
305
SERVER COPIES SELECTED THEME SETTINGS TO SETTING DOCUMENT FOR ROOM
306
SERVER SENDS NEW SET OF INSTRUCTIONS TO BROWSER TO REPRESENT DOCUMENT
307
OPTION 2: TWEEK EXISTING THEME
303
SERVER PROVIDES LIST OF OPTIONS TO TWEEK
308
USER SELECTS A USER INTERFACE (UI) ELEMENT TO TWEEK
309
SERVER DISPLAYS POSSIBLE TWEEKABLE ATTRIBUTES
310
USER SELECTS ATTRIBUTES
311
SERVER MODIFIES ROOM SETTINGS DOCUMENT
312
REFRESH
313

FIG. 31

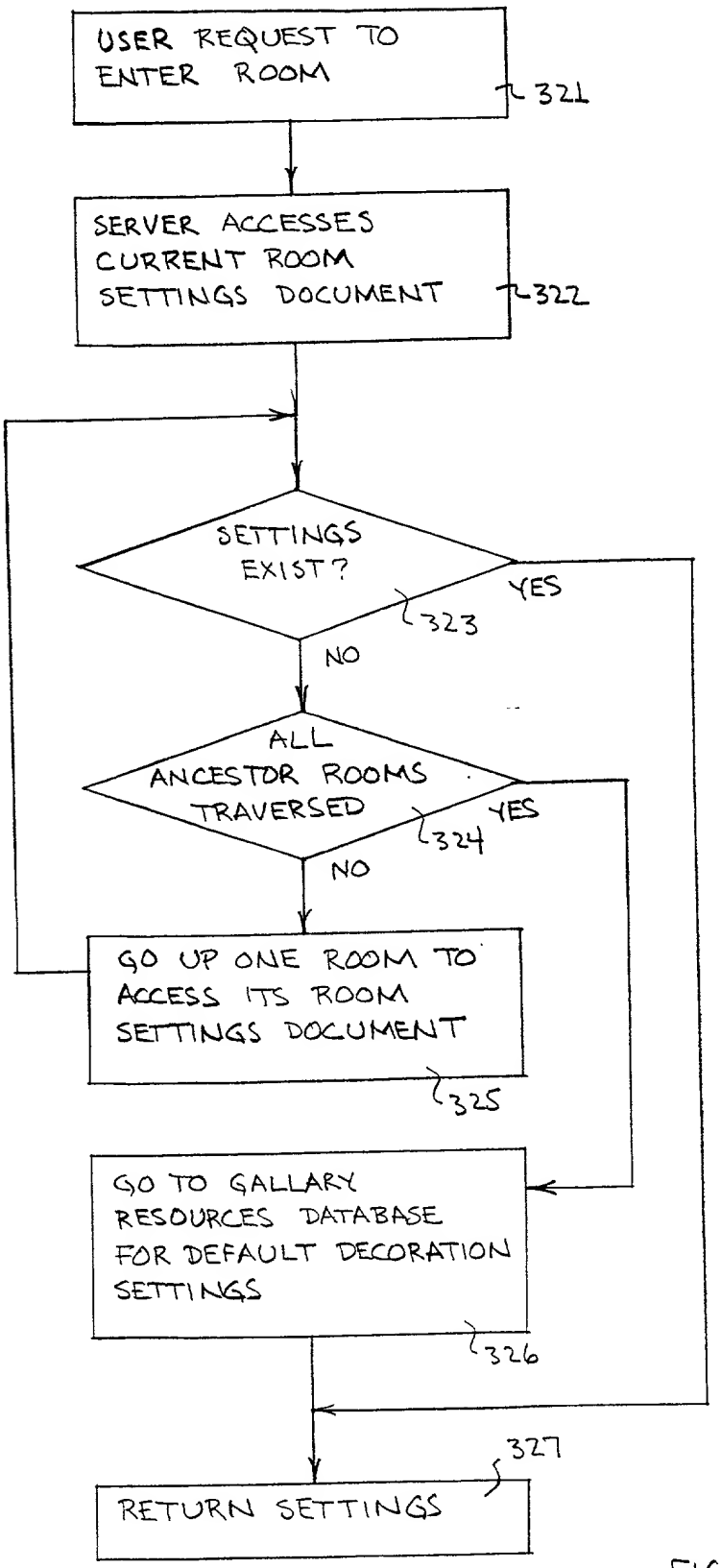

USER REQUEST TO ENTER ROOM
321
SERVER ACCESSES CURRENT ROOM SETTINGS DOCUMENT
322
SETTINGS EXIST?
YES
323
NO
ALL ANCESTOR ROOMS TRAVERSED
YES
324
NO
GO UP ONE ROOM TO ACCESS ITS ROOM SETTINGS DOCUMENT
325
GO TO GALLARY RESOURCES DATABASE FOR DEFAULT DECORATION SETTINGS
326
327
RETURN SETTINGS

FIG. 32

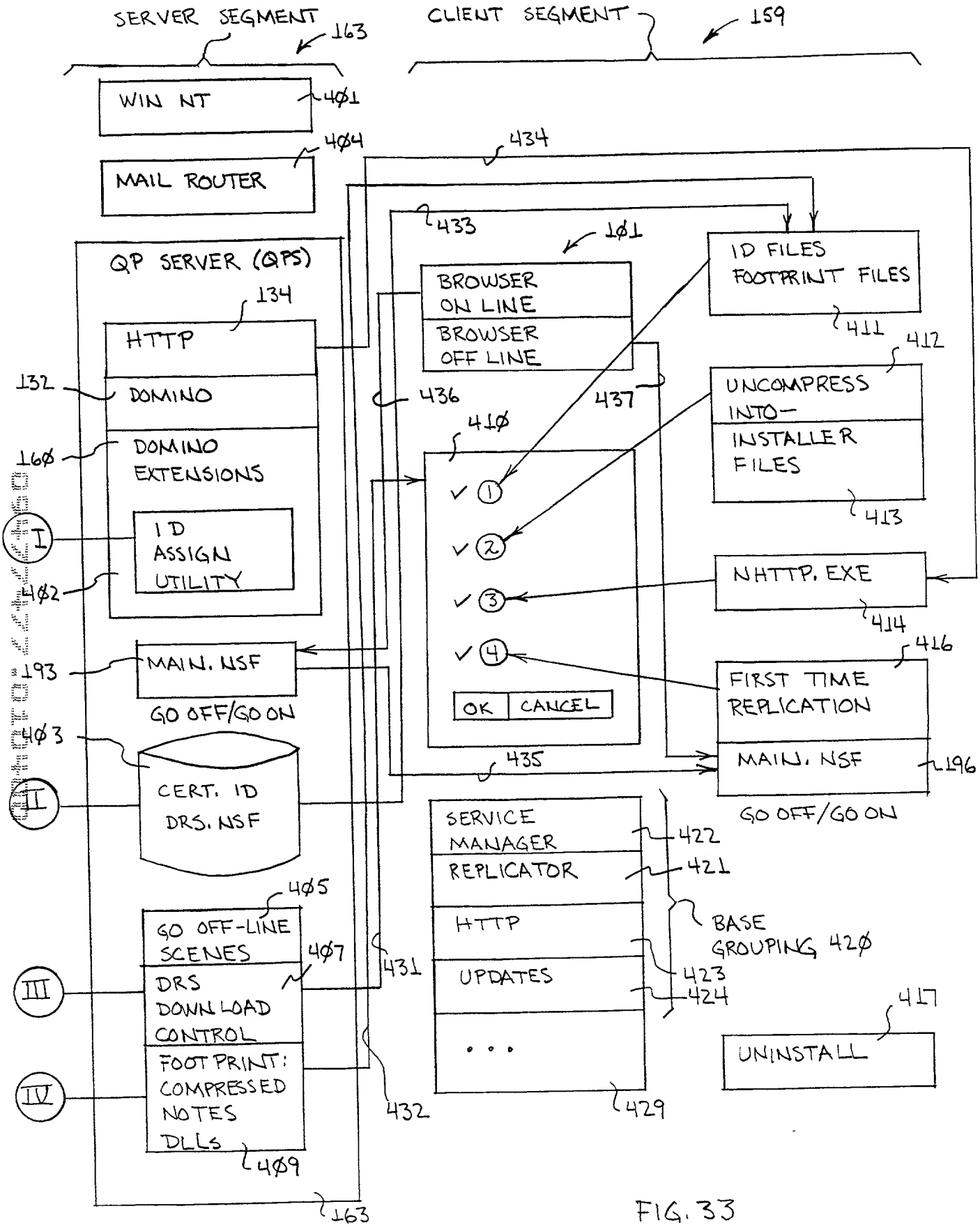

SERVER SEGMENT
163
CLIENT SEGMENT
159
WIN NT
401
MAIL ROUTER
404
434
433
101
QP SERVER (QPS)
134
HTTP
132
DOMINO
436
437
160
DOMINO EXTENSIONS
410
I
ID ASSIGN UTILITY
402
BROWSER ON LINE
BROWSER OFF LINE
ID FILES FOOTPRINT FILES
411
412
UNCOMPRESS INTO—
INSTALLER FILES
413
NHTTP.EXE
414
193
MAIN.NSF
GO OFF/GO ON
403
II
CERT. ID DRS.NSF
435
416
FIRST TIME REPLICATION
MAIN.NSF
196
GO OFF/GO ON
OK
CANCEL
SERVICE MANAGER
422
REPLICATOR
421
HTTP
BASE GROUPING 420
UPDATES
423
424
405
GO OFF-LINE SCENES
407
III
DRS DOWN LOAD CONTROL
431
IV
FOOT PRINT: COMPRESSED NOTES DLLs
409
432
429
417
UNINSTALL
163

FIG. 33

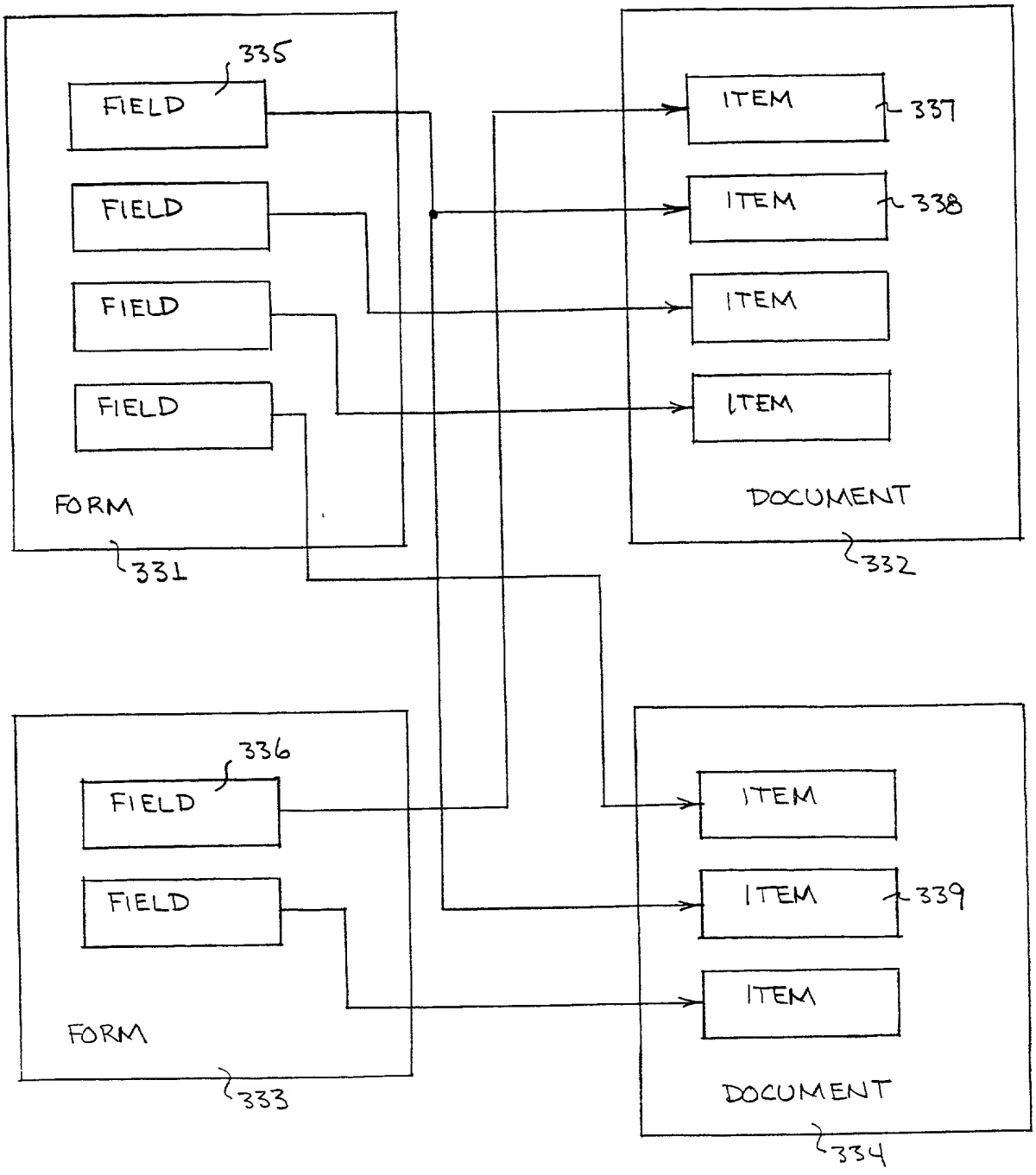

335
FIELD
FIELD
FIELD
FIELD
FORM
331
ITEM
337
ITEM
338
ITEM
ITEM
DOCUMENT
332
336
FIELD
FIELD
FORM
333
ITEM
ITEM
339
ITEM
DOCUMENT
334

FIG. 34

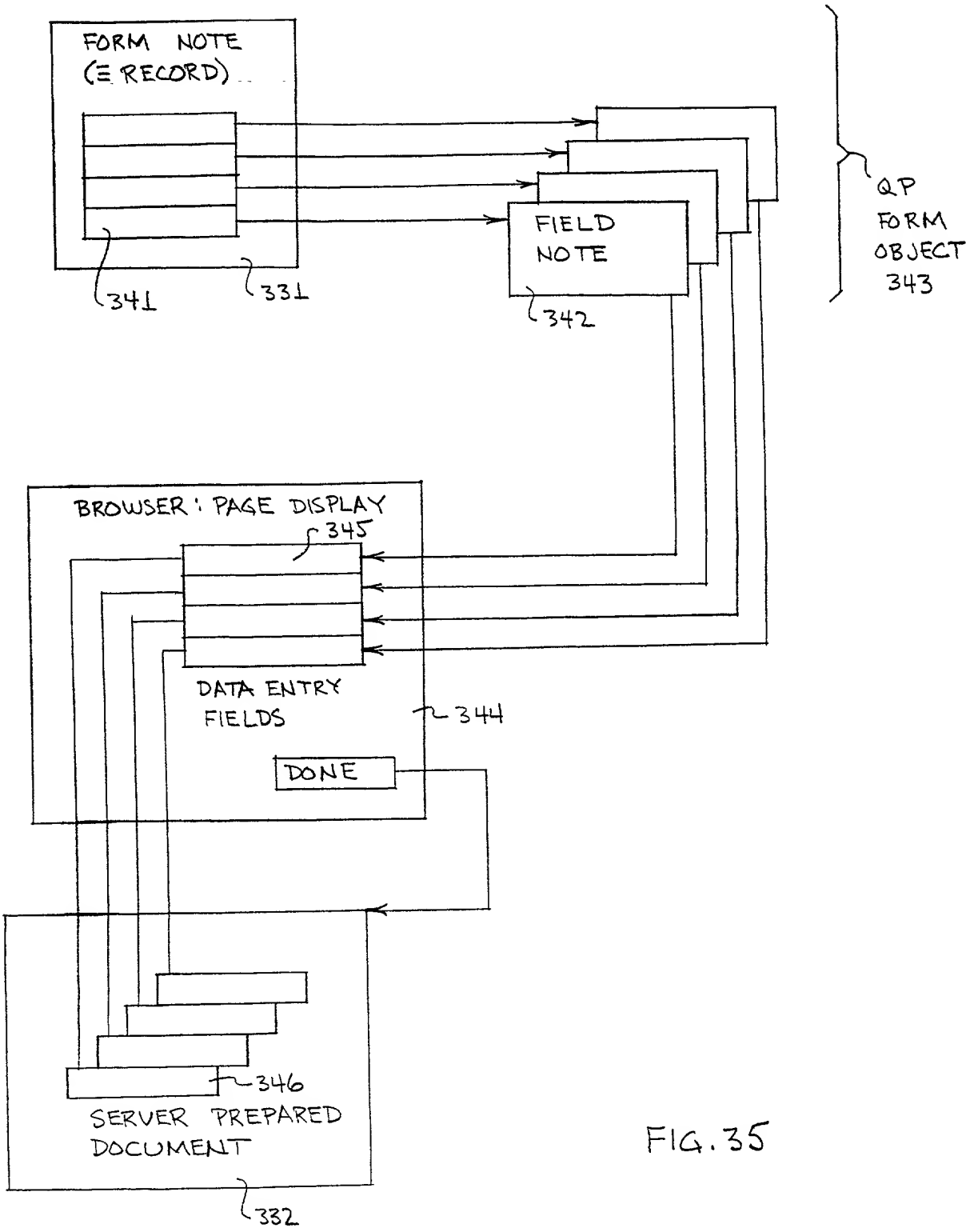

FORM NOTE
(≡ RECORD)
341
331
FIELD
NOTE
342
QP
FORM
OBJECT
343
BROWSER : PAGE DISPLAY
345
DATA ENTRY
FIELDS
344
DONE
346
SERVER PREPARED
DOCUMENT
332

FIG. 35

Practitioner's Docket No. L09-99-048 **PATENT**

COMBINED DECLARATION AND POWER OF ATTORNEY

(ORIGINAL, DESIGN, NATIONAL STAGE OF PCT, SUPPLEMENTAL, DIVISIONAL, CONTINUATION, OR C-I-P)

As a below named inventor, I hereby declare that:

TYPE OF DECLARATION

This declaration is of the following type:

(check one applicable item below)

☒ original.

☐ design.

☐ supplemental.

NOTE: If the declaration is for an International Application being filed as a divisional, continuation or continuation-in-part application, do not check next item; check appropriate one of last three items.

☐ national stage of PCT.

NOTE: If one of the following 3 items apply, then complete and also attach ADDED PAGES FOR DIVISIONAL, CONTINUATION OR C-I-P.

NOTE: See 37 C.F.R. § 1.63(d) (continued prosecution application) for use of a prior nonprovisional application declaration in the continuation or divisional application being filed on behalf of the same or fewer of the inventors named in the prior application.

☐ divisional.

☐ continuation.

NOTE: Where an application discloses and claims subject matter not disclosed in the prior application, or a continuation or divisional application names an inventor not named in the prior application, a continuation-in-part application must be filed under 37 C.F.R. § 1.53(b) (application filing requirements — nonprovisional application).

☐ continuation-in-part (C-I-P).

INVENTORSHIP IDENTIFICATION

WARNING: *If the inventors are each not the inventors of all the claims, an explanation of the facts, including the ownership of all the claims at the time the last claimed invention was made, should be submitted.*

My residence, post office address and citizenship are as stated below, next to my name. I believe that I am the original, first and sole inventor *(if only one name is listed below)* or an original, first and joint inventor *(if plural names are listed below)* of the subject matter that is claimed, and for which a patent is sought on the invention entitled:

TITLE OF INVENTION

SYSTEM AND METHOD FOR DYNAMICALLY GENERATING VIEWABLE GRAPHICS

SPECIFICATION IDENTIFICATION

the specification of which:

(complete (a), (b), or (c))

(a) ☒ is attached hereto.

NOTE: *"The following combinations of information supplied in an oath or declaration filed on the application filing date with a specification are acceptable as minimums for identifying a specification and compliance with any one of the items below will be accepted as complying with the identification requirement of 37 CFR 1.63:*

"(1) name of inventor(s), and reference to an attached specification which is both attached to the oath or declaration at the time of execution and submitted with the oath or declaration on filing;

"(2) name of inventor(s), and attorney docket number which was on the specification as filed; or

"(3) name of inventor(s), and title which was on the specification as filed."

Notice of July 13, 1995 (1177 O.G. 60).

(b) ☐ was filed on _____, as ☐ Serial No. 0 /_____
or ☐ _____
and was amended on _____ *(if applicable).*

NOTE: *Amendments filed after the original papers are deposited with the PTO that contain new matter are not accorded a filing date by being referred to in the declaration. Accordingly, the amendments involved are those filed with the application papers or, in the case of a supplemental declaration, are those amendments claiming matter not encompassed in the orginal statement of invention or claims. See 37 C.F.R. § 1.67.*

NOTE: *"The following combinations of information supplied in an oath or declaration filed after the filing date are acceptable as minimums for identifying a specification and compliance with any one of the items below will be accepted as complying with the identification requirement of 37 CFR 1.63:*

"(A) application number (consisting of the series code and the serial number, e.g., 08/123,456);

"(B) serial number and filing date;

"(C) attorney docket number which was on the specification as filed;

"(D) title which was on the specification as filed and reference to an attached specification which is both attached to the oath or declaration at the time of execution and submitted with the oath or declaration; or

"(E) title which was on the specification as filed and accompanied by a cover letter accurately identifying the application for which it was intended by either the application number (consisting of the series code and the serial number, e.g., 08/123,456), or serial number and filing date. Absent any statement(s) to the contrary, it will be presumed that the application filed in the PTO is the application which the inventor(s) executed by signing the oath or declaration."

M.P.E.P. § 601.01(a), 7th Ed.

(c) ☐ was described and claimed in PCT International Application No. _____, filed on _____ and as amended under PCT Article 19 on _____ *(if any).*

SUPPLEMENTAL DECLARATION (37 C.F.R. § 1.67(b))

(complete the following where a supplemental declaration is being submitted)

☐ I hereby declare that the subject matter of the

☐ attached amendment

☐ amendment filed on _____

was part of my/our invention and was invented before the filing date of the original application, above-identified, for such invention.

ACKNOWLEDGEMENT OF REVIEW OF PAPERS AND DUTY OF CANDOR

I hereby state that I have reviewed and understand the contents of the above-identified specification, including the claims, as amended by any amendment referred to above.

I acknowledge the duty to disclose information, which is material to patentability as defined in 37, Code of Federal Regulations, § 1.56,

(also check the following items, if desired)

☐ and which is material to the examination of this application, namely, information where there is a substantial likelihood that a reasonable Examiner would consider it important in deciding whether to allow the application to issue as a patent, and

☐ in compliance with this duty, there is attached an information disclosure statement, in accordance with 37 C.F.R. § 1.98.

PRIORITY CLAIM (35 U.S.C. §§ 119(a)–(d))

NOTE: "The claim to priority need be in no special form and may be made by the attorney or agent if the foreign application is referred to in the oath or declaration as required by § 1.63. The claim for priority and the certified copy of the foreign application specified in 35 U.S.C. 119(b) must be filed in the case of an interference (§ 1.630), when necessary to overcome the date of a reference relied upon by the examiner, when specifically required by the examiner, and in all other situations, before the patent is granted. If the claim for priority or the certified copy of the foreign application is filed after the date the issue fee is paid, it must be accompanied by a petition requesting entry and by the fee set forth in § 1.17(i). If the certified copy is not in the English language, a translation need not be filed except in the case of interference; or when necessary to overcome the date of a reference relied upon by the examiner; or when specifically required by the examiner, in which event an English language translation must be filed together with a statement that the translation of the certified copy is accurate." 37 C.F.R. § 1.55(a).

I hereby claim foreign priority benefits under Title 35, United States Code, §§ 119(a)–(d) of any foreign application(s) for patent or inventor's certificate or of any PCT international application(s) designating at least one country other than the United States of America listed below and have also identified below any foreign application(s) for patent or inventor's certificate or any PCT international application(s) designating at least one country other than the United States of America filed by me on the same subject matter having a filing date before that of the application(s) of which priority is claimed.

(complete (d) or (e))

(d) ☐ no such applications have been filed.

(e) ☐ such applications have been filed as follows.

NOTE: Where item (c) is entered above and the International Application which designated the U.S. itself claimed priority check item (e), enter the details below and make the priority claim.

PRIOR FOREIGN/PCT APPLICATION(S) FILED WITHIN 12 MONTHS (6 MONTHS FOR DESIGN) PRIOR TO THIS APPLICATION AND ANY PRIORITY CLAIMS UNDER 35 U.S.C. § 119(a)-(d)

COUNTRY (OR INDICATE IF PCT)	APPLICATION NUMBER	DATE OF FILING (day, month, year)	PRIORITY CLAIMED UNDER 37 USC 119
			☐ YES NO ☐
			☐ YES NO ☐
			☐ YES NO ☐
			☐ YES NO ☐
			☐ YES NO ☐

CLAIM FOR BENEFIT OF PRIOR U.S. PROVISIONAL APPLICATION(S)
(34 U.S.C. § 119(e))

I hereby claim the benefit under Title 35, United States Code, § 119(e) of any United States provisional application(s) listed below:

PROVISIONAL APPLICATION NUMBER	FILING DATE
_____ /_____	_____
_____ /_____	_____
_____ /_____	_____

CLAIM FOR BENEFIT OF EARLIER US/PCT APPLICATION(S) UNDER 35 U.S.C. § 120

☐ The claim for the benefit of any such applications are set forth in the attached ADDED PAGES TO COMBINED DECLARATION AND POWER OF ATTORNEY FOR DIVISIONAL, CONTINUATION OR CONTINUATION-IN PART (C-I-P) APPLICATION.

ALL FOREIGN APPLICATION(S), *IF ANY*, FILED MORE THAN 12 MONTHS (6 MONTHS FOR DESIGN) PRIOR TO THIS U.S. APPLICATION

_____

_____

_____

NOTE: If the application filed more than 12 months from the filing date of this application is a PCT filing forming the basis for this application entering the United States as (1) the national stage, or (2) a continuation, divisional, or continuation-in-part, then also complete ADDED PAGES TO COMBINED DECLARATION AND POWER OF ATTORNEY FOR DIVISIONAL, CONTINUATION OR C-I-P APPLICATION for benefit of the prior U.S. or PCT application(s) under 35 U.S.C. § 120.

POWER OF ATTORNEY

I hereby appoint the following practitioner(s) to prosecute this application and transact all business in the Patent and Trademark Office connected therewith.

(list name and registration number)

Stephen Keohane, Esq. Reg. No. 34,360

Shelley M Beckstrand Reg. No. 24,886

(check the following item, if applicable)

☐ I hereby appoint the practitioner(s) associated with the Customer Number provided below to prosecute this application and to transact all business in the Patent and Trademark Office connected therewith.

☒ Attached, as part of this declaration and power of attorney, is the authorization of the above-named practitioner(s) to accept and follow instructions from my representative(s).

SEND CORRESPONDENCE TO

☒ Address
Lotus Development Corporation
55 Cambridge Parkway
Cambridge, MA 02142

☐ Customer Number _____

DIRECT TELEPHONE CALLS TO:
(Name and telephone number)

Stephen Keohane, Esq.
(617) 693-4152

DECLARATION

I hereby declare that all statements made herein of my own knowledge are true and that all statements made on information and belief are believed to be true; and further that these statements were made with the knowledge that willful false statements and the like so made are punishable by fine or imprisonment, or both, under Section 1001 of Title 18 of the United States Code, and that such willful false statements may jeopardize the validity of the application or any patent issued thereon.

SIGNATURE(S)

NOTE: Carefully indicate the family (or last) name, as it should appear on the filing receipt and all other documents.

NOTE: Each inventor must be identified by full name, including the family name, and at least one given name without abbreviation together with any other given name or initial, and by his/her residence, post office address and country of citizenship. 37 CFR § 1.63(a)(3).

NOTE: Inventors may execute separate declarations/oaths provided each declaration/oath sets forth all the inventors. Section 1.63(a)(3) requires that a declaration/oath, inter alia, identify each inventor and prohibits the execution of separate declarations/oaths which each sets forth only the name of the executing inventor. 62 Fed. Reg. 53,131, 53,142, October 10, 1997,

Full name of sole or first inventor

Julio (*GIVEN NAME*) _____ (*MIDDLE INITIAL OR NAME*) Estrada (*FAMILY (OR LAST NAME)*)

Inventor's signature _____

Date _____ **Country of Citizenship** USA

Residence 67 Ledgeways, Carlisle, MA 01741

Post Office Address SAME

Full name of second joint inventor, if any

Miguel (*GIVEN NAME*) A. (*MIDDLE INITIAL OR NAME*) Estrada (*FAMILY (OR LAST NAME)*)

Inventor's signature _____

Date _____ **Country of Citizenship** USA

Residence 22 Crestwood Drive, Hollis, NH 03049

Post Office Address SAME

Full name of third joint inventor, if any

Joseph (*GIVEN NAME*) A. (*MIDDLE INITIAL OR NAME*) Russo (*FAMILY (OR LAST NAME)*)

Inventor's signature _____

Date _____ **Country of Citizenship** USA

Residence 11 Plain Road, Westford, MA 01886

Post Office Address SAME

Full name of sole or fourth inventor

Maurice	B.	Shore
(GIVEN NAME)	(MIDDLE INITIAL OR NAME)	FAMILY (OR LAST NAME)

Inventor's signature _____

Date _____ Country of Citizenship USA

Residence 26 Ash Street, Concord, MA 01742

Post Office Address _____

Full name of fifth joint inventor, if any

(GIVEN NAME)	(MIDDLE INITIAL OR NAME)	FAMILY (OR LAST NAME)

Inventor's signature _____

Date _____ Country of Citizenship _____

Residence _____

Post Office Address _____

Full name of ~~Sixth~~ joint inventor, if any

(GIVEN NAME)	(MIDDLE INITIAL OR NAME)	FAMILY (OR LAST NAME)

Inventor's signature _____

Date _____ Country of Citizenship _____

Residence _____

Post Office Address _____

(check proper box(es) for any of the following added page(s) that form a part of this declaration)

☒ **Signature** for fourth and subsequent joint inventors. *Number of pages added* _____

* * *

☐ **Signature** by administrator(trix), executor(trix) or legal representative for deceased or incapacitated inventor. *Number of pages added* _____.

* * *

☐ **Signature** for inventor who refuses to sign or cannot be reached by person authorized under 37 CFR 1.47. *Number of pages added* _____

* * *

☐ Added page for **signature** by one joint inventor on behalf of deceased inventor(s) where legal representative cannot be appointed in time. (37 CFR 1.47)

* * *

☐ Added pages to combined declaration and power of attorney for divisional, continuation, or continuation-in-part (C-I-P) application.

☐ Number of pages added _____

* * *

☐ Authorization of practitioner(s) to accept and follow instructions from representative.

* * *

(if no further pages form a part of this Declaration, then end this Declaration with this page and check the following item)

☒ This declaration ends with this page.